VINKTAR

Linux distribution for secure creation of cold storage wallets

Lucas Johns

March 28, 2019

Abstract

This paper describes the implementation of a live distribution for the creation of cold storage wallets. The aim of this implementation is to simplify the process of creating paper wallets under high security requirements. The source code as well as a current build of the distribution can be obtained at GitHub.

VINTKAR-LIVE
Source code: `github.com/envake/vinktar-live`
Current build: `github.com/envake/vinktar-live/releases/latest`

1 Introduction

In order to store crypto currencies securely, the storage of private keys in cold storage wallets has proven its worth. These are wallets that are never connected to the internet. The private keys are either stored on special hardware or can alternatively be printed on paper. The latter is called a paper wallet. Since the purchase of a hardware wallet is associated with costs, paper wallets are often used as cheap alternative for the long term storage of crypto currencies. To create these paper wallets, a valid key pair consisting of a private key and a public key must be generated. Tools have been available for a long time for all common crypto currencies, mostly implemented in JavaScript. Countless articles and tutorials explain how to create a Bitcoin paper wallet [1]. But there are also offers from scammers who have manipulated the key generation [2]. However, it is safer to generate the keys offline. Ultimately, generating private keys in a small Linux system running offline has proven to be an effective solution. Distributions such as Tails or Ubuntu Live-USB are mostly used. But these are in no way optimized for such a special task. The following sections describe a solution that enables a secure and simple paper wallet creation process in a Linux live system.

2 System structure

2.1 Targets of VINKTAR distribution

- minimal structure
- offline, no root
- open source, auditable
- cryptographically secure RNG-concept
- intuitive frontend
- high printer compatibility
- modifiable/extensible

2.2 System base

The Debian live-build framework is used to build the distribution [3]. The Debian Project has already announced in 2010 to develop the Debian kernel completely without proprietary firmware [4]. The distribution is therefore based on a 32-bit Debian GNU/Linux testing release, since the "stable" release is usually based on very old versions of applications. In Live-Build, the configuration of the distribution is done via an extensive directory structure.

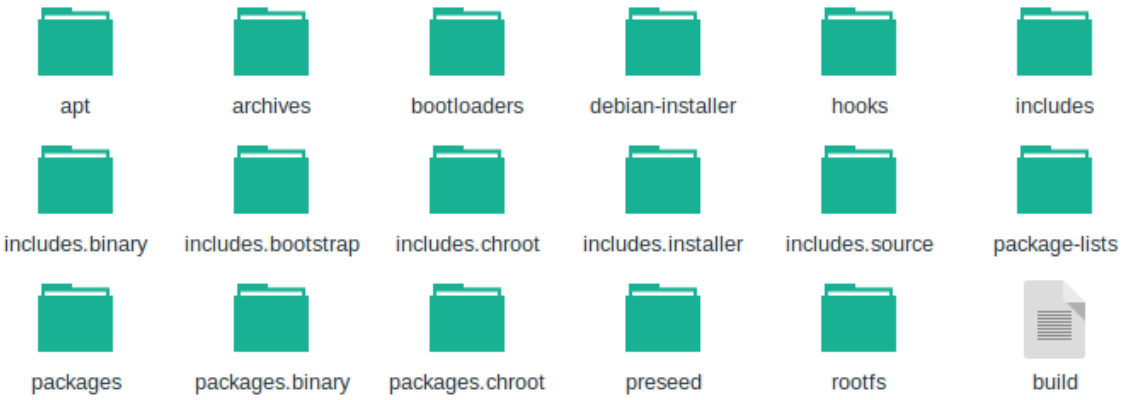


Figure 1: root configuration directory of live-build, image source: L. Johns

The entire system can be built from this structure. Packages contained in the system are organized in categorized `package-lists`. The system contains a browser, office programs and printer-related packages, which allow a high compatibility with common printer models.

Furthermore, user-defined scripts can be executed at certain times (hook-scripts). So-called 'includes' integrate custom files into the live distribution. This provides configuration files for the software contained in the system. The key generation tools are

Package list	Description	Example
apps.list.chroot	general applications	firefox-esr
desktop.list.chroot	xfce4-specific packages	xfce4-terminal
fonts.list.chroot	necessary fonts for included Software	fonts-liberation
live.list.chroot	necessary packages for live operation	user-setup
printing.list.chroot	printer-related setup and driver packages	system-config-printer
rng.list.chroot	rng-related packages	rng-tools

Table 1: Structure of the package lists with examples

also manually integrated into the file system. An essential software component of the distribution is the browser. Many key generation tools are implemented in JavaScript and run in the browser. Especially the cryptographic functions of the browser API become important for later use in the system. Mozilla Firefox ESR is used for this project. It is fully source-open and available in the Debian main repository. In general, the live system is designed to be very minimalistic. Both the included software and the user interface are focused on the use case. Further characteristics are the Init system systemd, the bootloader SYSLINUX and the used desktop Xfce4.



Figure 2: Xfce4-Desktop showing all included paper wallet tools; image source: L. Johns

The Xfce4 user interface is rebuilt from the ground up. A categorical assignment is used in the paper wallet menu. New applications to be integrated into the system only have to belong to the category `PaperWallets` and are displayed automatically. Furthermore, a theme with corresponding icons has been integrated for the general appearance. The distribution should be as intuitive as possible to use, which is why only usecase-relevant UI elements are available. A special feature of the system is that it starts by default with the root user account disabled. The Linux kernel can be started with the `noroot` option to prevent the use as root. The idea is that a root user account that cannot be used will not do any harm. Furthermore, the distribution is permanently disconnected from the network. This is also done using kernel parameters, since the separation should be made at the lowest point of the system stack.

3 Random number generator

Within the live system, secure random number generation is required. Currently, almost all of the tools included use the browser RNG functions. But there are also other implementations, such as a shell script for generating IOTA seeds. Using the Linux kernel RNG in a live environment requires further steps. Linux provides two device files as entropy source. These are `/dev/random` and `/dev/urandom`. This entropy pool is filled with noise values from the kernel environment, e.g. from device drivers [5]. Since a system boot without additional user interaction is highly predictable, the collected entropy is normally stored on disk when the system is shut down. Since a live system starts with exactly the same disk image every time, this step is not possible.

The problem of random numbers in live systems is not new even for this special case. The Tails operating system claims to be able to use cryptographic tools. The developers of Tails try to increase the seeding of the Linux kernel with two additional entropy sources [6]. Specifically, these are two daemons that are integrated into the system. The first is rngd, which uses an existing hardware RNG for seeding the entropy pool until a defined limit is reached. Not every system has a hardware RNG, so rngd does not help in every case. Therefore, there is a second source that is realized with the haveged daemon. This uses the HAVEGE algorithm (Hardware Volatile Entropy Gathering and Expansion), which collects entropy from unpredictable states of the processor [7]. It is not a replacement for a hardware RNG. The developer classifies the algorithm as follows.

> *“One could theoretically reproduce the sequence if he/she was able to reproduce all the past events on the machine. They are not pseudo-random either since there is no (short) seed which would allow an exact reproduction of the random sequence. The randomness results instead from an inability to control or predict with sufficient accuracy the events involved in the generation process.“* [8]

According to the description of the official Debian package, haveged is intended primarily for use on server and headless systems with limited user interaction [9]. Live systems such as Tails or the distribution developed here have in common with server systems that at the time of using random numbers only few user interactions took place. The use of haveged in virtualized environments is sometimes controversially discussed [10], the HAVEGE algorithm may not be a secure entropy source. The live distribution developed here is therefore generally not recommended for use in virtualized instances.

4 Integrated software

Initially, tools for 13 crypto currencies are planned. It is possible to extend this at any time. The selection of crypto currencies should appeal to as many users as possible. Therefore, the crypto currencies with the currently highest trading volume are selected [11].

Crypto currency	**Software**	**Implementation**	**RNG**
Bitcoin	bitaddress.org	JavaScript	crypto.getRandomValues, user generated entropy
Ethereum	myetherwallet	JavaScript	crypto.getRandomValues
Ripple XRP	ripply.eu	JavaScript	crypto.getRandomValues
Bitcoin Cash	bitcoin.com	JavaScript	crypto.getRandomValues, user generated entropy
EOS	eoscafe paper-wallet	JavaScript	crypto.getRandomValues
Stellar	stellar-paper-wallet	JavaScript	crypto.getRandomValues
Litecoin	liteaddress.org	JavaScript	crypto.getRandomValues, user generated entropy
Monero	monero-wallet-generator	JavaScript	crypto.getRandomValues, user generated entropy
Tron	tronpaperwallet.org	JavaScript	crypto.getRandomValues
IOTA	IOTA-Paper-Wallet	JavaScript	none, seed generation is implemented from kernel rng
Dash	paper.dash.org	JavaScript	crypto.getRandomValues, user generated entropy
NEO	Ansy	JavaScript	crypto.getRandomValues
Ethereum Classic	myetherwallet	JavaScript	crypto.getRandomValues

Table 2: Integrated software for cryptographic key generation

An exception of the selected tools is the software 'IOTA-Paper-Wallet'. Here the private key is not generated in the browser. The tool expects the input of the 81-digit IOTA seed, which is generated from the characters A-Z and 9 [12]. To simplify the use in the distribution, a short shell script is written for the generation of secure seeds, which is executed automatically when the browser is opened in an additional terminal window. In the script, five seeds are generated with the following shell command.

```
cat /dev/urandom |tr -dc A-Z9|head -c\${1: -81}
```

5 Typical usecase

In the following, a typical application example is constructed and explained step by step. Two paper wallets will be created and printed, one for Litecoin and one for EOS. This requires an image of the distribution and a USB stick with at least one gigabyte of storage. First the image is written on the stick. For a Windows system a tool like Rufus [13] is suitable. In Linux the shell program `dd` can be used. In this example Linux is used and the USB stick is located under `/dev/sdd`, so that the following command results.

```
sudo dd if="vinktar-live-image-i386.hybrid.iso" of="/dev/sdd"
```

The stick can now be connected to the computer where the live system is to be used. Next, the network cable should be removed. If there are manual switches for WLAN or Bluetooth, these can be switched off. The printer is also connected to the computer via USB. The computer is now started and should boot the live system. If this does not happen automatically, a specific key must usually be pressed to manually select the boot media during the boot process. In the bootloader menu, pressing Enter starts the system regularly. After only a few seconds the desktop of the live system appears. Clicking on the 'Paper Wallets' menu will list all crypto currencies for which the system can create paper wallets. In this case EOS is selected first. The browser opens and displays the generated keys including QR codes. The paper wallet can be printed by clicking on 'Print' or by entering `Ctrl` + `P`. If the printer is not listed here, it can usually be configured via the printer settings with 'Add'. Then it appears in the printer selection. For certain requirements on the layout of the paper wallet, the office programs contained in the system can also be used alternatively. For example, several keys can be printed on one page. Once the process is completed, the "Litecoin (LTC)" item is selected via the "Paper Wallets" menu. Since "liteaddress" includes the mouse movement as entropy source, the keys do not appear immediately. Otherwise the procedure is the same. Finally the system can be shut down. Additional options are disconnecting the system from its power source and resetting the printer to factory settings.

References

[1] **Alexander Weipprecht.** krypto-magazin.de. Bitcoin Paper-Wallet erstellen. [Online] 2018. `https://www.krypto-magazin.de/bitcoin-paper-wallet-erstellen/`, verfügbar am 14.11.2018

[2] **Mirko Ross.** heise.de. Kryptowährung: IOTA im Wert von vier Millionen US-Dollar geklaut. [Online] 2018. `https://www.heise.de/ix/meldung/Kryptowaehrung-IOTA-im-Wert-von-vier-Millionen-US-Dollar-geklaut-3952723.html`, verfügbar am 05.09.2018,15:08

[3] **Debian Live Team.** debian developers' corner. Debian Live Project. [Online] 2018. `https://www.debian.org/devel/debian-live/index.en.html`, verfügbar am 10.10.2018

[4] **Debian Projekt.** Debian Nachrichten. Debian 6.0 Squeeze wird mit vollständig freiem Linux-Kernel veröffentlicht. [Online] 2010. https://www.debian.org/News/2010/20101215, verfügbar am 27.09.2018

[5] **Michael Kerrisk** Ubuntu Manpage Repository. Ubuntu Manpage: random, urandom - Kernel-Geräte zur Erzeugung von Zufallszahlen. [Online] 2018. http://manpages.ubuntu.com/manpages/precise/de/man4/random.4.html, verfügbar am 01.09.2018, 11:35.

[6] **Tails project.** tails.boum.org. Tails - Random numbers. [Online] 2018. https://tails.boum.org/contribute/design/random/, verfügbar am 15.11.2018

[7] **Olivier Rochecouste.** irisa.fr. HAVEGE Hardware Volatile Entropy Gathering and Expansion, Overview. [Online] 2006. http://www.irisa.fr/caps/projects/hipsor/, verfügbar am 15.11.2018

[8] **Olivier Rochecouste..** irisa.fr. Execution time of a short sequence of instructions and hardware volatile states in a modern microprocessor. [Online] 2006. http://www.irisa.fr/caps/projects/hipsor/misc.php#measure, verfügbar am 15.11.2018

[9] **Debian Team.** packages.debian.org. Debian – Informationen über Paket haveged in buster. [Online] 2018. https://packages.debian.org/buster/haveged, verfügbar am 15.11.2018

[10] **Nic Waller.** security.stackexchange.com. Is it appropriate to use haveged as a source of entropy on virtual machines?. [Online] 2013. https://security.stackexchange.com/questions/34523/is-it-appropriate-to-use-haveged-as-a-source-of-entropy-on-virtual-machines, verfügbar am 16.11.2018

[11] **CoinMarketCap.** coinmarketcap.com. Cryptocurrencies Market Capitalization. [Online] 2018. https://coinmarketcap.com/, verfügbar am 19.10.2018

[12] **IOTA Foundation.** iota.org. Buy and Secure IOTA, Wallets and seeds. [Online] 2018. https://www.iota.org/get-started/buy-and-secure-iota, verfügbar am 16.11.2018

[13] **Rufus USB.** rufususb.com. Rufus - bootable USB flash drive. [Online] 2018. http://rufususb.com/, verfügbar am 13.11.2018