riscure

Software attacks on hardware wallets

Sergei Volokitin
Riscure, Netherlands

Introduction

Almost every security research has a question often left unanswered: what would be the financial consequence, if a discovered vulnerability is maliciously exploited? The security community almost never knows, unless a real attack takes place and the damage becomes known to the public. But now we have cryptocurrencies: a concept of digital money that is basically protected by a single private key, which, when stolen, leads to a measurable financial loss. Multiple breaches of private wallets and public currency exchange services are well-known, and to address the issue a few companies have come up with secure hardware storage devices to preserve the precious wallet access credentials at all costs.

But how secure are they? In this research, we show how software attacks can be used to break in the most protected part of the hardware wallet, the Secure Element, and how it can be exploited by an attacker.

The number of identified vulnerabilities in the hardware wallet show how software vulnerabilities in the TEE operating system can lead to a compromise of the memory isolation and a reveal of secrets of the OS and other user applications. Finally, based on the identified vulnerabilities an attack is proposed which allows anyone with only physical access to the hardware wallet to retrieve secret keys and data from the device. Additionally, a supply chain attack on a device allowing an attacker to bypass the security features of the device and have full control of the installed wallets on the device.

The specifics of Ledger Nano S

The hardware wallet based on dual chip design, which includes the STM32 micro-controller unit and secure element ST31. The MCU is used as “non-secure” part controlling USB communication, control buttons, device’s screen and communicating with the Secure Element. The secure element is the most protected part of the device with a certified IC designed to be resistant to physical attacks. The secure element has 320KB of flash memory which is used by the BOLOS TEE (trusted execution environment) and the wallets (user applications) to store the code and data including most critical assets like private keys and sensitive user data. Most of the code running on the device is open-source except the TEE OS code. The device is protected with a PIN code set up by a user on the first boot and in case the user PIN entered three times incorrectly the device will be wiped and allow to configure a new PIN.

Insufficient checks

The first four weaknesses discovered in the Ledger Nano S refer to errors in the device operation that can be used to reveal a certain portion of the device's firmware and user data, which was meant to be protected. Depending on how you look at it, these can be called vulnerabilities or just programming mistakes, because they do not lead to an immediate compromise of secrets meant to be protected. For example, vulnerability V1 allows dumping 8 kilobytes of firmware via dereferencing the null pointer provided as an input to a system call.

The table below shows results for a few calls of the function with different parameters:

PTR	LEN	OUTPUT – HASH SHA256
0x00000000	0x00000001	6e340b9cffb37a989ca544e6bb780a2c78901d3fb33738768511a30617afa01d
0x00000000	0x00000002	2ee788372518190a6ab539cbb20331df1040f21846e3ba836c269aee907c894c
0x00000000	0x00000003	df236376becfe951a5a3dfa7c274ed26a75f1ccba7cf432772a9cc349017eaac
0x00000000	0x00000004	5060868c58cbb70948e570f613517144e4072f31355d4a12e4e5257398511b5e
0x00000000	0x00000005	de41aa507bea243205cdc828096ddbca02d7f13d31fea5164abab9e2baf21ef1
0x00000000	0x00000006	a0337a5446cce751335a6cd98eb8e045a28e15ae390956aa3d9b1fab32374491

Although memory at the address 0x00000000 should not be accessible to userspace applications, the system call returns hash in this case instead of a security exception, giving an attacker a way of revealing the memory by brute-forcing byte by byte. The memory at the zero address seems to be write protected since any modification attempt fails, preventing furtherer exploitation of the vulnerability.

Vulnerabilities V2, V3 and V4 lead to a partial disclosure of a device's memory, including ROM, flash and RAM memory, via certain system calls. The vulnerabilities can be used in order to learn more about the memory of other wallet applications and the operating system and reduce the entropy of the secret key. The vulnerable system calls could be used as oracles which, given an arbitrary address, would return different exceptions depending on the content of the memory at the address. In case an attacker knows the location of another wallet application's secret key, he can reduce the entropy of the key by using the system call to get knowledge about the key byte values. Again, this does not lead to complete loss of secrets, but could be analyzed further and, perhaps in combination with other attack methods, help to break the security protection. In a device that is position as a secure storage for highly valuable data, even these code errors have to be eliminated.

Breaking application isolation

Ledger Nano S implements a Trusted Execution Environment (TEE) on the secure element which includes the operating system itself and wallet applications sharing the flash memory of the secure element. The operating system has a higher privilege level than the wallet applications and the code and data of the operating system has to be isolated from the installed wallet application. The wallet designed in such a way that third parties can easily create wallet applications for new cryptocurrencies and share the application with users to install the applications on their hardware wallets which makes the device more versatile and at the same time more vulnerable since an attacker can create a malicious wallet

application which accesses other wallet applications or the OS and steals their secrets. To ensure this is not possible wallet applications have to be isolated from each other and the operating system.

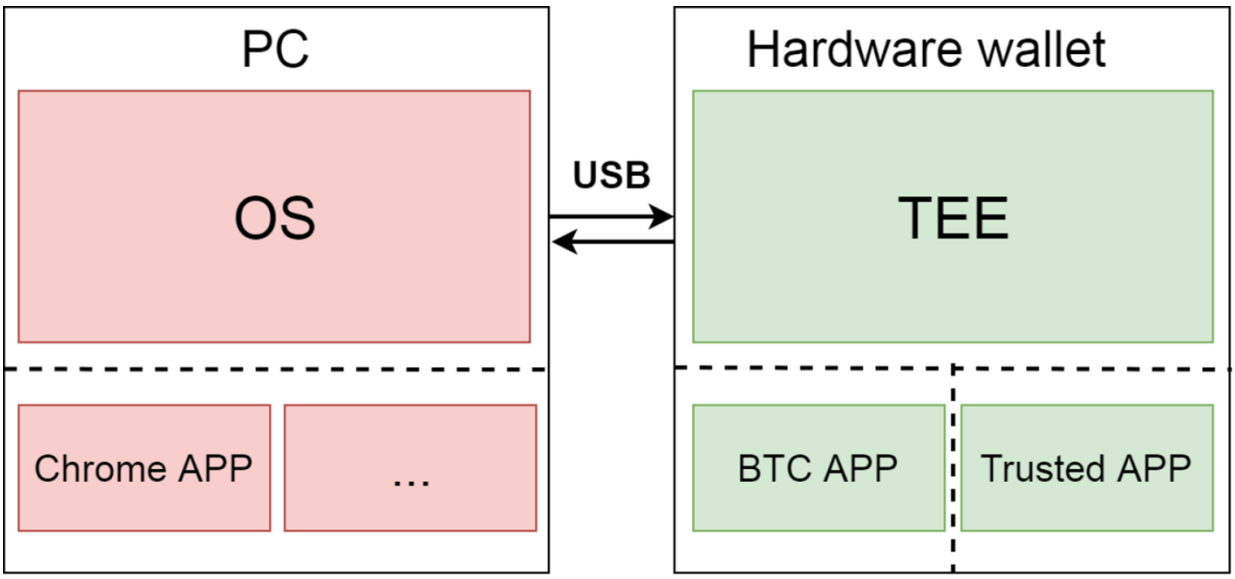

Fig. 1 - TEE design of the hardware wallet

Vulnerability V5 breaks the TEE security by installing a user application with a debug flag. When a debug flag is set, seemingly because of misconfigured memory protection unit, the application can read data from other secure containers as shown in Figure 2.

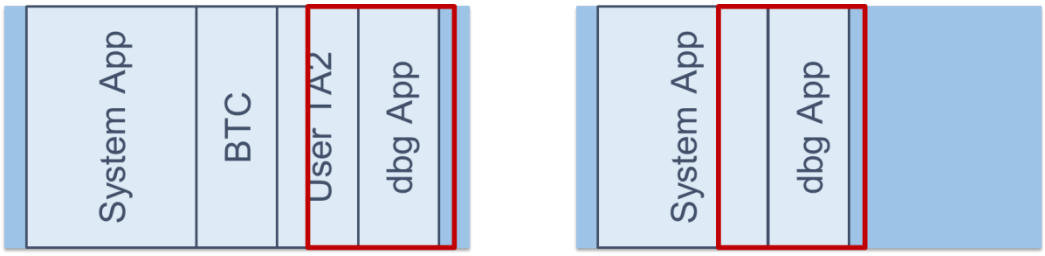

Fig.2 – Memory access of dbgApp to the flash memory

Unlike vulnerabilities V1-V4, this one is directly exploitable. When exploited by an attacker – for example, an adversary that has gained physical access to a device – it leads to a full compromise of certain apps. Fortunately, not all of them: the way some applications were designed makes some of them less affected by the vulnerability, for instance, the private key of the Bitcoin wallet application remained safe, but the vulnerability allows stealing the secret key from a Monero digital wallet as well as U2F application HMAC key, designed to secure access to online services.

Fig.3 – Source code of the applications storing secrets on the flash

In fact, this vulnerability can be utilized not only in a device theft scenario. It was discovered that resetting the wallet does not clear the secure user data (vulnerability V6). On a Ledger Nano S the wallet is reset or locked when a wrong access PIN is entered three times. The vendor assumes that resetting a device by entering the wrong PIN allows a user to dispose of or resell a digital wallet. In fact, the reset procedure erases the BIP39 passphrase used to derive private key. In some cases (Bitcoin storage app) it is enough to protect the private key even if it is preserved in an encrypted form after a reset. But a combination of V5 and V6 allows an attacker to steal a user's Monero wallet or U2F key after the device was reset.

Compromising the supply chain

The final finding V7 allows an attacker to install a rogue application on a Ledger Nano S before a device reaches its owner. Once secrets have been uploaded to a digital wallet, the rogue application can potentially be used to steal sensitive information, even without a physical access to a device. The rogue app could be a modified Bitcoin app with malicious functionality. In a normal situation modifying the code of an application leads to a clear warning displayed on a screen of a digital wallet.

Fig. 4 – Request to proceed executing app without manufacturers signature

What we have found is that Ledger allowed enrollment of CustomCA keys to simplify the development process. Self-signed app, therefore, does not display a warning, which enables the installation of a rogue app before sending a device to a customer (either by reselling a used digital wallet online or attacking the retail supply chain in another way). The issue, in this case, is caused by the fact that when the device is reset and wiped the CustomCA keys are not deleted and the apps are kept on the flash.

Conclusions

The vulnerabilities were disclosed to the manufacturer and security update 1.4 introduced the fixes of the vulnerabilities. This research does not try to replace a full security audit of a device but meant to show how vulnerabilities in the software of a device can be exploited by an attacker to get access to the secrets protected by the secure hardware. The aspects of hardware security were not even considered in this research, and, for a secure storage of sensitive information, physical attacks, despite the cost, might be used by an attacker to break the security.