Ledger Stax
Security Target

Release 1.2

LEDGER

Table of contents

1 Document Identification

1.1 Security Target Identification

Identification	Ledger Stax Security Target
Release	Version 1.2
Date	2024-12-03
Diffusion	Public

1.2 Security Target History

Release	Date	Author	Role	Comments
1.2	2024-12-01	Pierre HOTTELART	Security Certification Engineer	- Updated Mapping between assets and security functions
1.1	2023-12-01	Romain MUGUET	Security Certification Engineer	- Updated Embedded OS version to 1.1.0 - Updated security function 5 to detail side-loading - Updated the list of assumptions
1.0	2023-01-18	Romain MUGUET	Security Certification Engineer	Initial Release

1.3 Security Target Review

Release	Date	Reviewer	Role
1.2	2024-12-03	Raphaël GESLAIN	Embedded Software Director
1.2	2024-12-03	Michael MOUCHOUS	Hardware Security Researcher Manager
1.2	2024-12-03	Vincent BOUZON	Security Certification Director

2 Introduction

2.1 Document Context

This document constitutes the security target of the Ledger Stax in the context of a CSPN evaluation.

2.2 Documentation Identification

2.2.1 ANSSI related documents

The following tables identify the documents regarding the CSPN evaluation.

Reference	Title	Version	Date
[CER-P-01]	Certification de sécurité de premier niveau des produits des technologies de l'information	4.0	2022-03-03
[CRY-P01]	Modalités pour la realisation des analyses cryptographiques et des evaluations des générateurs de nombres aléatoires	4.1	2020-01-26
[PG-83]	Guide des mécanismes cryptographiques: Règles et recommandations concernant le choix et le dimensionnement des mécanismes cryptographiques	2.04	2020-01-01
[RGS_B2]	Règles et recommandations concernant la gestion des clés utilisées dans les mécanismes cryptographiques	2.0	2012-06-08
[RGS_B3]	Règles et recommandations concernant les mécanismes d'authentification	1.0	2010-01-13

2.2.2 Bitcoin Improvement Proposal

Reference	Title	Date
[BIP32]	Hierarchical Deterministic wallets	2020-11-04
[BIP39]	Mnemonic code for generating deterministic keys	2013-09-10
[BIP44]	Multi-Account Hierarchy for deterministic Wallets	2014-04-24
[SLIP-44]	Registered coin types for BIP-0044	2014-07-09

2.2.3 LEDGER

Reference	Title
[CMD]	Cryptographic Mechanisms Description, version 1.4
[CTM]	Cryptography Testing Methodology, version 1.0
[CTP]	Cryptographic Test Plan, version 1.0
[UM]	User Manual - Ledger Stax
[Ledger Live]	Ledger Live
[Embedded App Security]	Ledger Developer Portal - Embedded App Security
[Build & Load App]	Ledger Developer Portal - Build & Load App

Reference	Title
[Security Audits]	Ledger Developer Portal - Security Audits
[LEDGERCtl]	A Python library to control Ledger devices
[Get Started]	Set up your Ledger Stax

2.2.4 Certicom Research

Reference	Title	Version	Date
[SEC_2]	Certicom Research Standards for Efficient Cryptography SEC 2: Recommended Elliptic Curve Domain Parameters	2.0	2010-01-27

2.2.5 Federal Office for Information Security (BSI)

Reference	Title	Version	Date
[AIS31]	Functionality classes and evaluation methodology for physical random number generators	1.0	2001-09-25

2.2.6 STMicroelectronics Main Hardware

Reference	Type	Role
[ST33K1M5C]	Secure IC	Main Hardware offering an EAL 6+ security level as stated in [ST33_CC]
[STM32WB35]	MCU	Supporting Hardware
[ST25R3916]	NFC chip	Supporting Hardware
[ST33_CC]	Certification	ST33K1M5C Evaluation Assurance Level 6+ Certification report CC-21-0252712

3 Product Description

The Ledger Stax is a Personal Security Device (PSD) designed to securely store cryptographic secrets and provide cryptographic primitives. It also provides a secure cryptographic storage making it a platform on which the End-User can install applications, called Embedded Apps, thus extending the product usage from a hardware wallet to a Universal 2nd Factor token for example or even a password manager.

3.1 Operational Environment

The Ledger Stax is not a standalone product in the sense that it will be used in conjunction with web services. So to help the End-User and provide a smooth experience LEDGER developed [Ledger Live] software. Through [Ledger Live] the End-User is then able to install Embedded Apps on his Ledger Stax.

The diagram below illustrates the main interactions between elements when the [Ledger Live] is required:

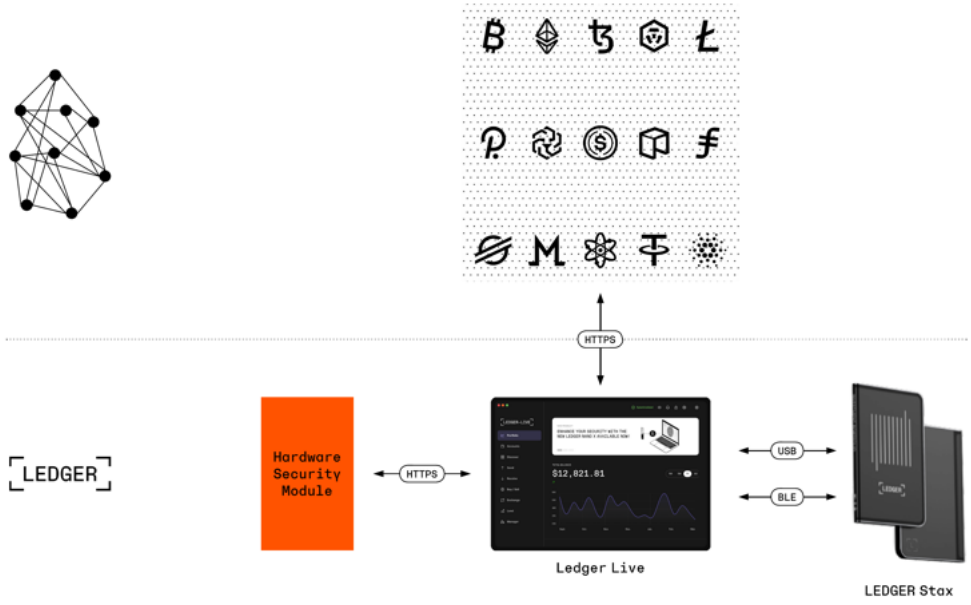

Figure 1: Environment with Ledger Live

There are also use cases where [Ledger Live] is not needed, such as when the Ledger Stax is used as a U2F token or in conjunction with a software cryptocurrency wallet such as MetaMask. The following diagram illustrates the main interactions between elements when the [Ledger Live] is not required:

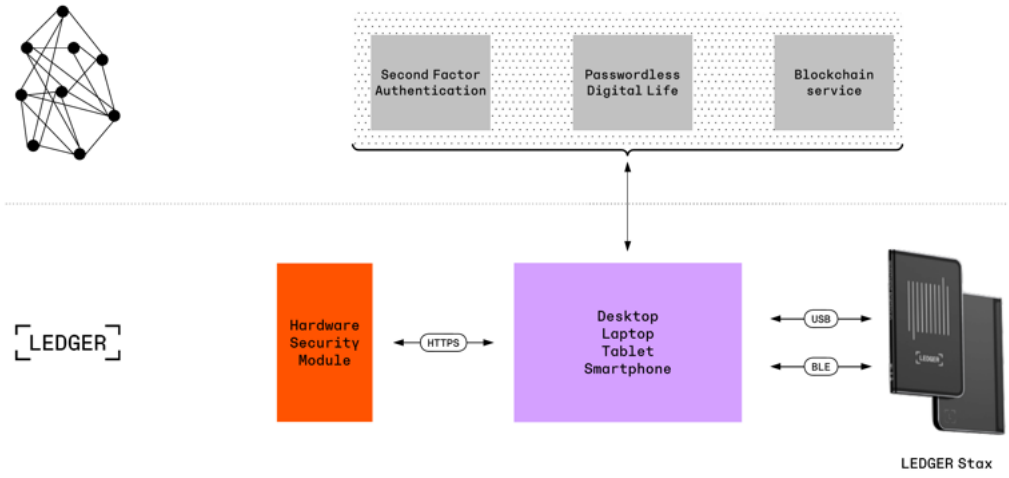

Figure 2: Environment without Ledger Live

3.2 Ledger Stax Features

The Ledger Stax supports the following features:

- Multi-services: Hardware Wallet, NFT visualiser, Cryptographic Platform, Password Manager, Second Factor authenticator (FIDO)
- Comply with several cryptocurrencies: **Bitcoin**, **Ethereum**, Solana, Polkadot, etc. [1]
- USB connectivity
- BLE v5.0 connectivity
- Piezoelectric speaker (hereinafter referred to as piezo)
- NFC connectivity: a personalised NFC tag (url, txt) can be defined so that a NFC reader can bump into Ledger Stax and read it. No reader mode is implemented by the Ledger Stax
- Open Source Embedded App: all Embedded Apps developed by LEDGER can be reviewed and verified by End-Users (e.g. Bitcoin, Ethereum)
- Developer friendly: develop a Embedded App and then install it on the Ledger Stax.
- Comply with the main BIP standards: [BIP32], [BIP39] and [BIP44]
- Multi-platform: 64-bits desktop computer (Windows 10+, macOS 12+ and Ubuntu LTS 20.04+ (excluding ARM Processors) and smartphones (iOS 13+ or Android 8.1+)
- **E-ink screen**: to verify the transaction data (amount, address), to print a QR code or display a picture
- **Touch screen**: used to get the End-User's consent relative to sensitive operations like unlocking the device or processing a transaction
- **PIN**: to unlock the Ledger Stax
- **Plausible deniability**: an additional PIN linked to a passphrase can be defined to create an hidden account
- **Genuine PSD**: cryptographic attestation mechanisms ensuring that the Ledger Stax is a genuine one
- **Post-issuance capability**: all piece of software (Embedded OS, firmware and Embedded Apps) can be securely updated

[1] Supported Crypto Assets: https://www.ledger.com/supported-crypto-assets

Bold features are included in the security scope and addressed by dedicated security functions. The LEDGER's security model is based on the Secure Element technology and the embedded software developed by LEDGER. In other words, compromising the connectivity (either USB or BLE) or the Host application does not compromise the PSD.

3.3 Ledger Stax Architecture

The Ledger Stax is based on an architecture leveraging two hardware chips with each a specific set of tasks:

- A generic MCU: general purpose chip
- A Secure IC: handling all security related tasks

3.3.1 The generic MCU

The MCU, considered as a supporting hardware, is in charge of:

- Communicating with the Host via USB or BLE
- Communicating with the Secure IC
- Communicating with the NFC chip
- Communicating with the piezo
- Monitoring the battery level
- Handling the battery management

The MCU used by the Ledger Stax is the [STM32WB35].

3.3.2 The Secure IC

The Secure IC, with the code running on it, is in charge of properly protecting the End-User's assets and for the handling of all sensitive operations. Therefore it is in charge of (but not limited to):

- Generating the seed
- Deriving the corresponding Key Pair
- Signing transactions
- Communicating with the MCU
- Driving the screen
- Receiving the notifications from the touch screen

For this specific reason, the Ledger Stax uses the [ST33K1M5C] microcontroller, which belongs to the Secure Element Technology. This microcontroller is also Common Criteria certified (refer to [ST33_CC] to get further details), making the Ledger Stax a secure hardware wallet.

LEDGER's security model for the Ledger Stax is based on the use of this dual-chip architecture. The Host, the BLE or NFC communication could be compromised without compromising the overall security of the Ledger Stax. Also the End-User actively participates in the security: all sensitive operations must get the End-User's consent (PIN validation, transaction confirmation) achieved via the touch screen.

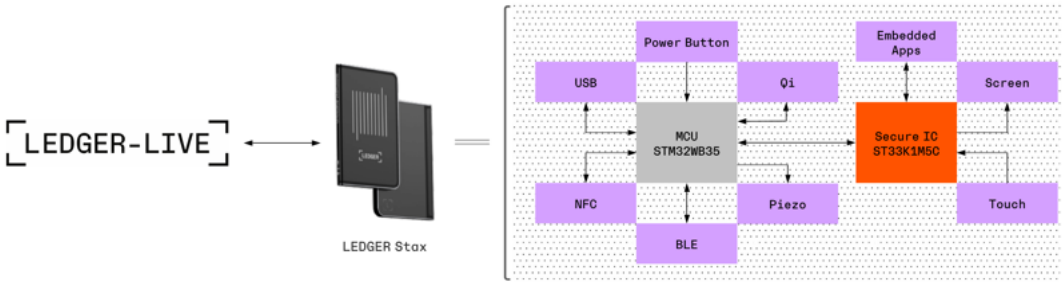

Figure 3: Zoom on the Ledger Stax

4 Product Use Cases

The object of this section is to put forward the use cases that enter the scope of the evaluation. By doing so, LEDGER aims at linking the End-User's actions with the actual security functions implemented in the Ledger Stax and/or its environment. Therefore making it clearer for the End-User to understand how LEDGER secures the use of its products (see also Mapping between Use cases and Security Functions).

The following sections summarises the uses cases defined in the Ledger Stax User Manual ([UM]) that enter the scope of the evaluation. As they are hereafter summarised, please refer to the [UM] for more details.

4.1 Setting Up a New Ledger Stax[2]

As reminded in [Get Started]:

> *"If a user were to receive a device containing a pre-completed recovery phrase or a PIN code, the user should not use the device, as it means that the device may have already been used by somebody else. LEDGER will never provide a PIN code or recovery phrase with the product, nor ever ask for them. Under these circumstances, the user must contact LEDGER customer support."*

The following steps are mandatory to properly initialize a new device:

1. **Checking for factory settings**:
   When the End-User powers on his Ledger Stax for the first time, it should display the LEDGER logo and a sentence (refer to section "Check factory settings" of [Get Started]). Also it should not be preconfigured with a PIN code.
2. **Set up your Ledger Stax with [Ledger Live] (Optional step)**:
   The End-User installs [Ledger Live] either on a computer or a smartphone. For the latter, Ledger Stax will present a QR code on the screen so that the End-User's smartphone can open or download the [Ledger Live] Mobile app. With a computer, the USB cable shall be used.
3. **Set up a PIN code**:
   The PIN unlocks your device and the Embedded Apps on it. A 8-digit PIN code offers an optimal level of security.

[2]This setup phase is also referred to as "*Onboarding*".

4. **Set up your Ledger Stax as a new device**:
   It will generate new private keys so you can manage your crypto assets. You will also write down a new 24-word recovery phrase, also called mnemonic seed, the only backup of your private keys.
5. **Or, restore your device from a recovery phrase**:
   It will recover the private keys linked to an existing recovery phrase.

For advanced End-Users, it is also possible to manually[3] check the Secure Element attestation, proving that it has been manufactured by LEDGER. You can verify it by running:

```
pip install --no-cache-dir ledgerblue
python -m ledgerblue.checkGenuine --targetId 0x33200004
```

See [Python Loader Installation] and [Python Loader Exploitation] for more information.

4.2 Using a Ledger Stax

The following use cases are the ones a End-User will encounter on everyday basis:

6. **Unlocking your device**:
   After powering on his device the End-User is requested to enter the PIN he has previously chosen during the set up step. Be warned that after three incorrect PIN code entries, the device will reset to factory settings, erasing the private keys from their secure storage.
7. **Embedded App installation**:
   The End-User will need to install Embedded Apps on his device to manage different crypto assets or to transform his device into a Universal 2nd Factor token. These Embedded Apps can be installed with [Ledger Live] software (see Operational Environment), where an App catalog is available. The End-User can also side-load Embedded Apps directly on his device.
8. **Adding an account**:
   The End-User will have to add an account for each crypto asset he wants to manage. [Ledger Live] helps you doing such action. By adding an account, the device will use the seed and a specific derivation path, unique for each crypto asset, to generate a public key for this specific crypto asset.
9. **Sending crypto assets**:
   The End-User can send crypto assets from his accounts in [Ledger Live] to a recipient address. The private key used to sign the transaction is derived from the seed on a specific derivation path, unique for each crypto asset. He is then required to use his device to verify and approve the transaction.
10. **Address verification**:
    When verifying an address with Ledger Stax, you have the options to show the address within a QR code. This QR code is generated when you choose to show it, it's not an image stored on the device.

[3] The Secure Element attestation verification is also performed automatically. For example when an End-User downloads an Embedded App with [Ledger Live], a Secure Element attestation verification will then take place.

4.3 Updating the Ledger Stax

Over time the End-User will have to update the Embedded Apps as well as the Embedded OS and firmware.

11. **Updating the Embedded OS and firmware**:
    New features and security updates will be made available for the device. It is important that the End-User has the latest Embedded OS and firmware installed. The installation is done through [Ledger Live]. The e-ink screen's firmware can also be securely updated over time. Also, updating your device has no impact on your crypto assets or the functionality of your device.
12. **Updating Embedded Apps**:
    Through [Ledger Live] the End-User can update an Embedded App. But what actually happens is not an update but a deletion of the current version of the Embedded App and the installation of the newer version. Therefore the updating step is equivalent to the Embedded App installation step.

5 Target of Evaluation

5.1 Identification of the Evaluated Product

The following table identifies the Ledger Stax according to the CSPN process:

Product Name	Ledger Stax
Product category	Hardware and embedded software
Developer	LEDGER, 1 rue du Mail, 75002 Paris
Website	www.ledger.com
Embedded OS Evaluated	1.1.0
Bootloader version	0.44
Microcontroller version	5.19
Product reference	TargetID: 0x33 0x20 0x00 0x04

5.2 Target of Evaluation Scope

The security model created by LEDGER is based on the Secure Element technology. This Secure Element embeds a set of hardware security countermeasures (for instance active shield, monitoring of environmental parameters, True Random Number Generator [AIS-31] compliant).

Nevertheless, in order to get a product resistant against high attack potential, LEDGER has also implemented a set of software security countermeasures. It is the composition of hardware security mechanisms (provided by the Secure IC) and the software security mechanisms (provided by LEDGER) which make the Ledger Stax resistant against sophisticated attacks.

The Target of Evaluation, focused on the Ledger Stax, is identified in the following diagram:

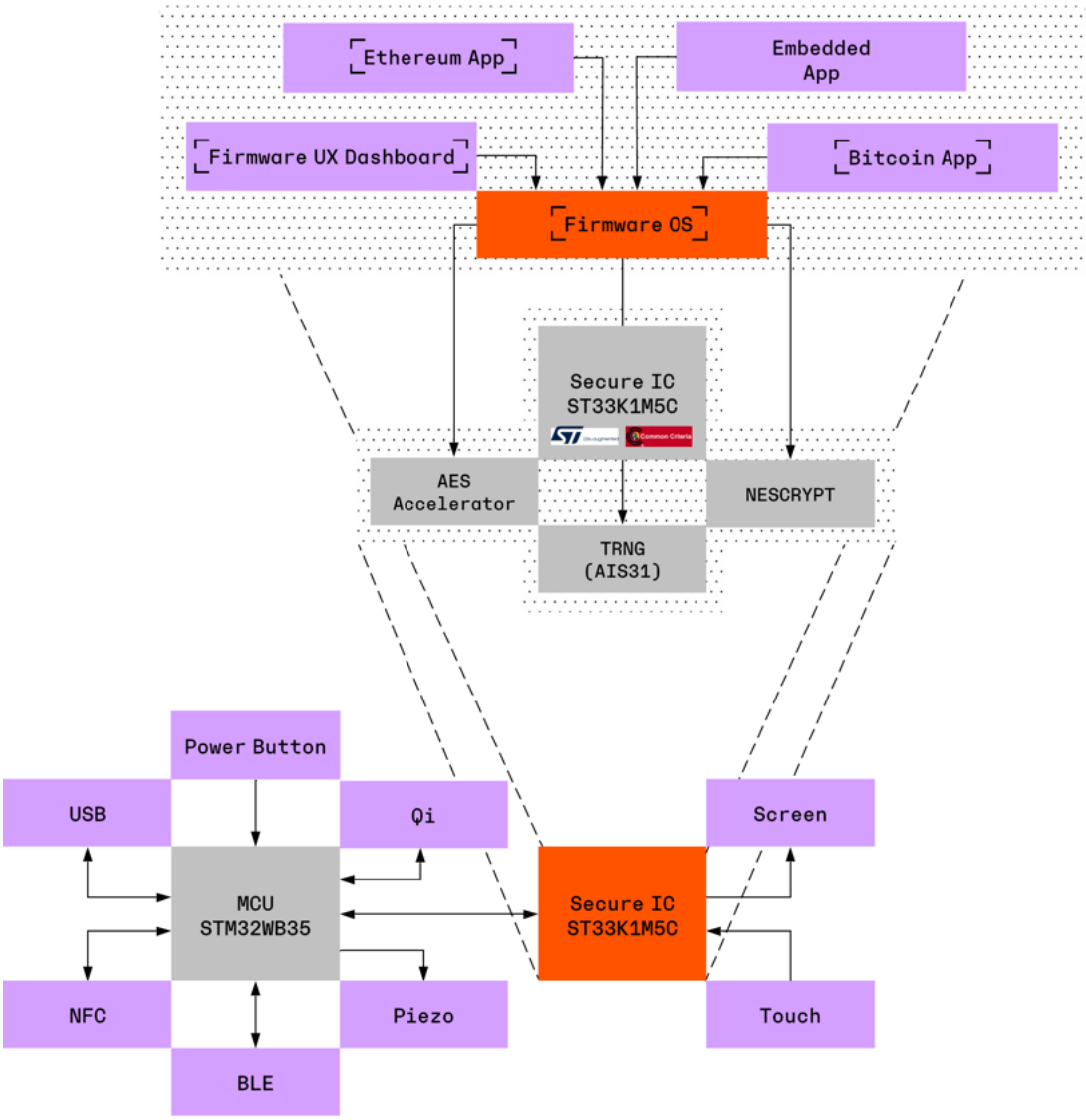

Figure 4: Target of Evaluation including a zoom on the SE

The ToE includes:

- Physical elements
  - One button
  - One e-ink screen
  - One USB-C port
  - One piezo
- Wireless interfaces
  - Bluetooth Low Energy
  - NFC
  - Qi wireless charging
- Hardware
  - MCU v5.19 (see the generic MCU section)

  - Secure IC (see the Secure IC section)
- DVT4
  - Main Board rev 4.0
  - BT FPC rev 1.0
  - QiNFC FPC rev G11
- Software developed and secured by LEDGER
  - The firmware running on top of the MCU
  - The Embedded OS running on top of the secure IC
  - The Embedded App labelled *Dashboard App* executed by the Embedded OS

5.2.1 MCU Firmware

The firmware is in charge of:

- Communicating with the outside world through USB, BLE or NFC
- Communicating with the Embedded OS

5.2.2 The Embedded OS

Embedded OS is in charge of:

- Managing peripherals (screen and touch interface)
- Performing cryptographic computations
- Storing secret data (seed, PIN)
- Offering a set of API (communication, cryptographic primitives, seed) accessible to all Embedded Apps

5.2.3 The Dashboard App

The Dashboard App, launched as soon as the device is started, is in charge of:

- Verifying the End-User's PIN
- Setting up a new Ledger Stax: seed generation and PIN enrollment
- Presenting all the installed Embedded Apps for the End-User to select
- Managing the Embedded Apps: installation and deletion

The Dashboard App ensures a UX consistency whatever the running Embedded App is, by managing for instance the screen.

5.2.4 Reference Embedded Apps

For the evaluation Ledger has decided to use the Bitcoin App and Ethereum App as references.

It's essential to note that these aforementioned applications do not fall within the scope of evaluation as they do not implement any security function outlined in this security target. Their presence is solely for the laboratory's use during the evaluation process and to highlight certain security function (see Security Function #5: App Isolation).

5.3 Assumptions

Below is the list of assumptions:

1. The Ledger Stax is acquired from the official Ledger website, official Amazon stores[4] or an official LEDGER reseller
2. If a user were to receive a device containing a pre-completed recovery phrase or a PIN code, the user shall not use the device, as it means that this latter may have already been used by somebody else
3. The HSM is properly operated by LEDGER
4. The LEDGER engineering team working on the Embedded OS, Embedded Apps, the firmware and the HSM are competent, trained and non-hostile
5. The End-User has verified that the Ledger Stax has not been tampered
6. The Ledger Stax is either powered off or locked (i.e., an End-User verification is required) when the PSD is either stolen or found
7. The scenario where the Ledger Stax is stolen, to be tampered with, and then handed back to its End-User (otherwise called "return theft") is not part of the evaluation scope
8. All techniques consisting in spying the End-User's interactions with the PSD are out of the scope. This covers for example a CCTV focused on the Ledger Stax and any other more sophisticated attack

5.4 Environment Measures

Even if the Ledger Stax can be used within a strict environment (for instance storing the device inside a vault, signing a transaction inside a secure building), the security design developed by LEDGER allows the End-User to experience the PSD in a public area. The device is architectured to provide an high assurance level to the End-User whatever the environment.

Nevertheless, with the Ledger Stax security model, the security is shared between the Ledger Stax device and the End-User. The following security rules must be fulfilled:

- The End-User **must** maintain the PIN secret
- The End-User **must** keep the 24-word recovery phrase secret
- The End-User **must** ensure that no one has access to the recovery sheet
- The End-User **must** verify transactions' details and address displayed on the screen are valid

5.5 End-User

One of LEDGER's ambitions is to ensure that the blockchain technology can be appropriated by all. While the solution is sophisticated, the Ledger Stax offers a simple and natural User Experience. The Ledger Stax being designed to be a mainstream technology, is user-friendly and can be easily manipulated by an End-Users with no technical background.

5.6 Assets

As the PSD processes sensitive operations (i.e., sign transactions, manage passwords, achieve U2F authentication, ...) and stores confidential data, the following assets must be secured:

1. Random Number (Data)
2. Secret Seed (Data)
3. PSD Keys:
   - PSD.PublicKey (Data)
   - PSD.PrivateKey (Data)

[4] Official Amazon stores in the USA, Australia, Canada, United Kingdom, Germany, France, Spain, Italy, the Netherlands, Poland, Sweden, Turkey, and Japan.

- PSD.PublicKey.Sig (Data)
- PSD.Ephemeral.PrivateKey (Data)

4. HSM Keys:
   - HSM.PublicKey (Data)
   - HSM.Ephemeral.PublicKey (Data)
5. Session Keys:
   - ECDH.Secret (Data)
   - ENC.Session.key (Data)
   - MAC.Session.key (Data)
6. PIN Data:
   - Reference PIN (Data)
   - PIN Try Counter (Data)
   - PIN Try Limit (Data)
   - PIN Result (Data)
7. Secret Data protected by the PIN (Data)
8. Embedded OS (Data)
9. HSM.Ephemeral.Certificate Verification (Operation)
10. PIN Verification (Operation)
11. UPDATE.PublicKey (Data)
12. NENC.key (Data)

All the assets listed above are worth of interest to an adversary and are subject to a set of threats as mentioned in Threats.

6 Threat Description

6.1 Threat Agents

As one of the Ledger Stax features is to sign digital transactions, it is therefore considered as a sensitive device. This signature operation, used to unlock the cryptocurrency funds located on the blockchain, involves the manipulation of the private key. The owner of this private key (the End-User) is the owner of the corresponding cryptocurrency funds.

Additionally, and as described in the Product Description section, the Ledger Stax is not only a hardware wallet but also provides a set of added-value services. The Password Manager and FIDO are typical Embedded Apps making the digital End-User life frictionless and more secured.

Several threats are applicable to the Ledger Stax and can be divided into two classes:

1. Physical threats:
   The threat agent has a physical access to the Ledger Stax. This occurs when the Ledger Stax has been either stolen or found. The PSD's state is either powered off or locked when the PSD is either stolen or found. This PSD's state requires the PIN to get access to the sensitive services.
2. Remote threats:
   The threat agent has no physical access. This remote threat class is considered when the End-User's Host or BLE connection has been compromised. It is through the use of crafted USB or Bluetooth frames that the adversary will launch his attacks (i.e., signing a transaction, getting passwords).

The following section describes the main threats applicable to the Ledger Stax related to the two threat classes.

6.2 Threat #1: Generating a Biased or a Deterministic Random Number

Context:

The Random Number Generator included in the Ledger Stax is used to:

1. Generate a Random Number exploited as a seed
2. Participate in establishing a secure channel between the Ledger Stax and LEDGER's HSM

The Ledger Stax, compliant with [BIP32], is a deterministic hardware wallet. This feature indicates that a seed is generated by the device during the initialization. From this seed, the End-User has the capability to derive all Key Pairs required to manage the crypto assets accounts.

Note that this feature allows to recover the crypto assets funds if the Ledger Stax is lost, stolen or destroyed as long as the seed is correctly backed up (via the Recovery Sheet, see restore your device from a recovery phrase).

The Ledger Stax uses the Random Number Generator not only for generating the seed but also for creating a Secure Channel between the LEDGER's Secure Server and the Ledger Stax avoiding replay attacks.

Threat:

The entropy is the key element regarding a Random Number feature. The entropy must be ensured by a true random number. The main threat is to reduce the entropy so that it reduces dramatically the seed space. This seed's space size is 2^{256}.

If the sensitive number generation operation is controlled by the adversary, he is then able to predict all data which uses the seed to derive new secrets, such as the End-User's account, his crypto address, secure channel keys, etc.

6.3 Threat #2: Using an Ungenuine Ledger Stax

Context:

LEDGER is the unique manufacturer of the Ledger Stax device. The authenticity proves the Ledger Stax is only issued by LEDGER avoiding some security holes related for instance to supply chain attacks. Besides, as the Ledger Stax is a sensitive device, it must only work as specified by LEDGER. For instance, the LEDGER embedded software, including not only Embedded OS but also a set of Embedded Apps, must be executed as expected.

In other words, both authenticity and integrity of the Ledger Stax must be ensured.

Threat:

The main threats are:

1. Manufacturing a fake Ledger Stax: as an adversary manufactures a fake Ledger Stax, it has the full control on the device and can create malicious Ledger Stax.
2. Modifying the Ledger Stax produced by LEDGER: an adversary adds malicious software or hardware to dump out sensitive data.

6.4 Threat #3: Unwanted Access to the Ledger Stax

Context:

The Ledger Stax embeds a set of sensitive services. One of them is related to the management of crypto assets. For instance, an End-User can create through his Ledger Stax a set of accounts (i.e., a professional account, an individual account, a family account) linked to several cryptocurrencies (i.e., Bitcoin, Ethereum). The Embedded Apps installed on the Ledger Stax can sign transactions to unlock the funds or transfer an NFT for example.

Note that the Ledger Stax offers several sensitive services (FIDO, Password Manager) interesting for an adversary as well.

Threat:

The main threat is related to a threat agent having physical access to the End-User's Ledger Stax to perform unwanted action such as sending crypto assets or accessing the End-User's FIDO protected accounts.

This threat is also applicable remotely when the End-User's Host has been previously compromised.

6.5 Threat #4: Compromising the Post-Issuance Capability

Context:

The Ledger Stax includes a post-issuance capability making possible to update not only Embedded Apps but also the Embedded OS and MCU firmware. This feature, giving LEDGER an incredible flexibility, can be exploited to:

1. Add new services

2. Fix some functional and protocol issues
3. Reinforce the security of the Ledger Stax

Threat:

The main threat is to take advantage of the update functionality to inject a malicious firmware or Embedded OS so that an adversary can take the full control of the Ledger Stax.

6.6 Threat #5: Application Impersonation

Context:

The Ledger Stax are Hierarchical Deterministic (HD) wallets (see [Terminology] and [BIP32]), meaning that they can derive different cryptographic secrets from a single seed.

Threat:

The LEDGER devices use the industry standards [BIP32] and [BIP44] to derive cryptographic secrets from a single seed. A malicious Embedded App, once installed on the Ledger Stax, can derive all possible HD path.

This way, if for example the Ledger Stax had previously generated an account for Bitcoin, the malicious Embedded App could retrieve the private key and impersonate the Bitcoin App to then steal the bitcoins.

7 Security Functions

As raised in the previous section, the Ledger Stax can be targeted with four main threats. These threats are critical because they can compromise the Ledger Stax: deterministic random number, access to the device without End-User verification, fake or malicious Ledger Stax.

LEDGER has developed appropriate security functions explained in this section to properly block each threat. It is worth highlighting that the implementation of these security functions relies on a set of security mechanisms. This security methodology of adding several security layers (defence-in-depth concept) counteracts not only straightforward but also sophisticated attacks.

7.1 Security Function #1: True Random Number Generator

Description:

This security function #1, labelled True Random Number Generator, aims at counteracting threat #1.

This security function is based on the TRNG embedded in [ST33K1M5C]. This TRNG, evaluated according to [AIS-31] methodology, has been successfully certified Class PTG.2.

To reinforce the entropy of the generated Random Number, LEDGER has also implemented an additional software post-processing countermeasure.

Assets:

The assets related to security function #1 are:

1. Random Number - entropy - (data)
2. Secret Seed (data)

7.2 Security Function #2: Attestation Mechanism

Description:

This security function #2, labelled Attestation Mechanism, aims at blocking threat #2.

LEDGER has implemented a solution to ensure that any Ledger Stax is a genuine:

- This solution is based on a Public Key Infrastructure (based on secp256k1 elliptic curve), with LEDGER as the Certification Authority. This dedicated infrastructure, based on Hardware Security Module, is not only administrated but also operated by LEDGER.
- It starts in the factory when all LEDGER devices are provisioned, they first generate a unique device public-private keypair. The device's public key is then signed by LEDGER's Issuer key to create an Issuer Certificate which is stored in the device. This certificate is a digital seal of authenticity of the LEDGER device. By providing the device's public key and Issuer Certificate, the device can prove that it is a genuine LEDGER device.

Then, when the Ledger Stax is connected to the Host and under some circumstances (for instance an Embedded App download, Embedded OS or MCU firmware installation), a mutual authentication between the Ledger Stax and LEDGER's HSM is performed.

This security function #2 relies on the following commands:

1. VALIDATE_TARGET_ID
2. INITIALIZE_AUTHENTICATION
3. VALIDATE_CERTIFICATE_LAST

4. GET_CERTIFICATE
5. GET_CERTIFICATE_LAST

The VALIDATE_TARGET_ID and INITIALIZE_AUTHENTICATION commands initiate the mutual authentication. The VALIDATE_CERTIFICATE_LAST command is used by the PSD to authenticate the HSM[5] while the GET_CERTIFICATE and GET_CERTIFICATE_LAST command is used by the HSM to authenticate the PSD.

At the end of this command/response sequence, a mutual authentication is achieved. Besides, both HSM and PSD have generated ephemeral keys leveraged during an ECDH to share a common secret only known by them.

This attestation mechanism is performed through a set of ECDSA operations as illustrated in the following diagram:

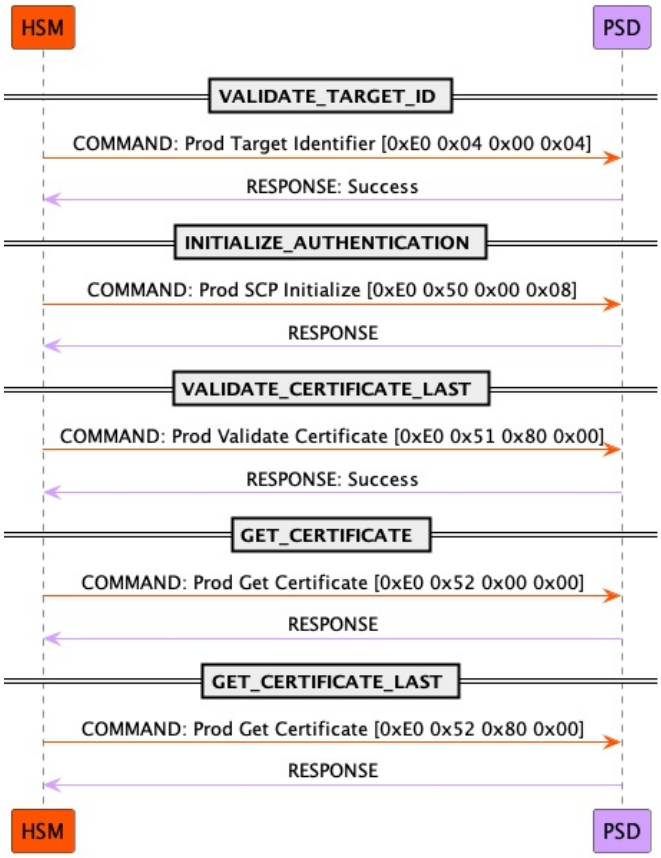

Figure 5: Security Function #2 - Attestation Mechanism

Assets:

The assets related to security function #2 are:

1. Random Number (Data)
2. PSD.PublicKey.Sig (Data)

[5] A VALIDATE_CERTIFICATE command exists but is not needed here. Indeed, during production in factory the PSD has stored a signature of its public key, as mentionned in the Description section above.

3. PSD.PublicKey (Data)
4. PSD.Ephemeral.PrivateKey (Data)
5. PSD.PrivateKey (Data)
6. HSM.Ephemeral.PublicKey (Data)
7. HSM.Ephemeral.Certificate verification (Operation)
8. HSM.PublicKey (Data)

7.3 Security Function #3: End-User Verification

Description:

This security function #3, labelled End-User Verification, aims at counteracting threat #3.

As soon as the Ledger Stax is connected to a Host, the End-User must prove that he is the owner of this Ledger Stax. This security function #3 is the first interaction between the End-User and the Ledger Stax. This security function is critical because it gives access to all services supported by the Ledger Stax.

The End-User verification is performed through a PIN verification. As a reminder, this PIN is defined by the End-User during the the Set up a PIN code step. Also while defining the PIN he defines its length, which has to be in the following range: minimum 4 digits, maximum 8 digits.

The End-User directly enters the PIN value using the touch screen. This candidate PIN is then compared to the Reference PIN stored in the SE. A correct verification allows the End-User to use all services provided by the Ledger Stax. For instance, all cryptocurrency Embedded Apps are then available meaning cryptocurrency transfer is available. Note that all other Embedded Apps (for instance Password Manager, FIDO) are also only available as soon as the PIN verification is successfully performed.

The PIN Try Counter (PTC), whose default value is set to 3, counteracts brute-force attacks revealing the value of the PIN. As soon as the PTC exceeds its limit, the Ledger Stax wipes the following sensitive assets:

1. PIN
2. Seed
3. Secret Data protected by the PIN

Thanks to this security action of wiping, the Ledger Stax cannot be used because the current state is not operational anymore. A device setup is then required (see Setting up a new Ledger Stax).

A correct End-User verification unlocks all the Ledger Stax services and resets the PTC to 3.

Assets:

Several sensitive assets are used to ensure the End-User verification:

1. Secret data protected by the PIN (Data)
2. Reference PIN (Data)
3. PIN Try Counter (Data)
4. PIN Verification (Operation)
5. PIN Try Limit (Data)
6. PIN Result (Data)

7.4 Security Function #4: Post-Issuance Capability Over a Secure Channel

Description:

This security function #4, labelled Post-Issuance Capability over a Secure Channel, aims at counteracting threat #4.

This security function uses security assets generated during security function #2 execution (mutual authentication between the HSM and the PSD).

The Post-Issuance Capability over a Secure Channel is illustrated in the following diagram:

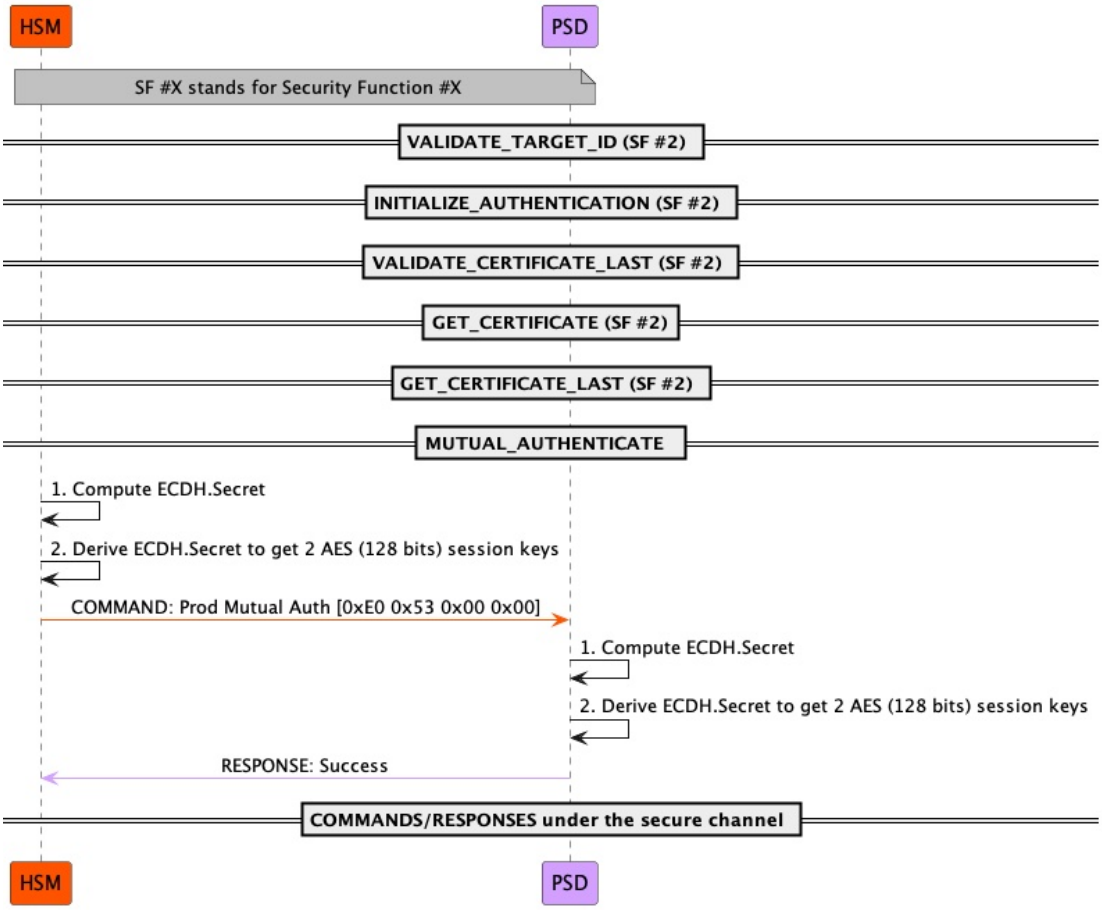

Figure 6: Security Function #4 - Secure Channel

The first commands (VALIDATE_TARGET_ID, INITIALIZE_AUTHENTICATION, VALIDATE_CERTIFICATE_LAST, GET_CERTIFICATE_LAST) performs a mutual authentication (security function #2) to ensure the HSM and PSD are genuine. Note that during the execution of the previous commands, both HSM and PSD have generated ephemeral EC key pairs. These ephemeral key pairs are leveraged to process an ECDH so that both HSM and PSD share a common secret labelled ECDH.Secret.

This ECDH.Secret is then derived to get 2 session keys:

- ENC.Session.Key
- MAC.Session.Key

These 2 session keys ensure the confidentiality and the integrity of messages (command/response) over the secure channel.

After the successful processing of the MUTUAL_AUTHENTICATE command, all following commands (secured in confidentiality and integrity) are managed inside the Secure Channel.

The secure channel is designed to block typical attacks. For instance, the secure channel does not accept the same set of commands twice, making replay attacks not operational anymore. Additionally, thanks to the current OS verification process, software installations are always upgrades.

It is not possible to downgrade the software version already installed on the `Ledger Stax`. This anti-rollback security protection discards all attack vectors related to install a previous software version containing a set of vulnerabilities already identified.

Assets:

1. Embedded OS (Data)
2. ECDH.Secret (Data)
3. ENC.Session.Key (Data)
4. MAC.Session.Key (Data)

7.5 Security Function #5: App Isolation

Description:

This security function #5, labelled App isolation, aims at counteracting threat #5.

One of the main features of `LEDGER` devices is that the End-User can load any Embedded App on his device, not only the ones coming form the `LEDGER` App catalog. It is then the Embedded OS that will provide an isolation between the Embedded Apps themselves, as well as an isolation between the Embedded Apps and the Secret Seed.

`LEDGER` has implemented the following rules:

- Embedded Apps cannot access the OS memory.
- Embedded Apps cannot read or write the volatile and non-volatile memory from another app.
- Embedded Apps can derive keys on their own Hierarchical Deterministic (HD) path only, which ensures that cryptocurrency apps cannot steal keys from each other.

7.5.1 Memory Protection

For the two-first rules, the Embedded OS relies on hardware features provided by the Secure Element, either the MPU (Memory Protection Unit) or the MMU (Memory Management Unit),

to isolate the apps between them and also to isolate the OS itself from the apps.

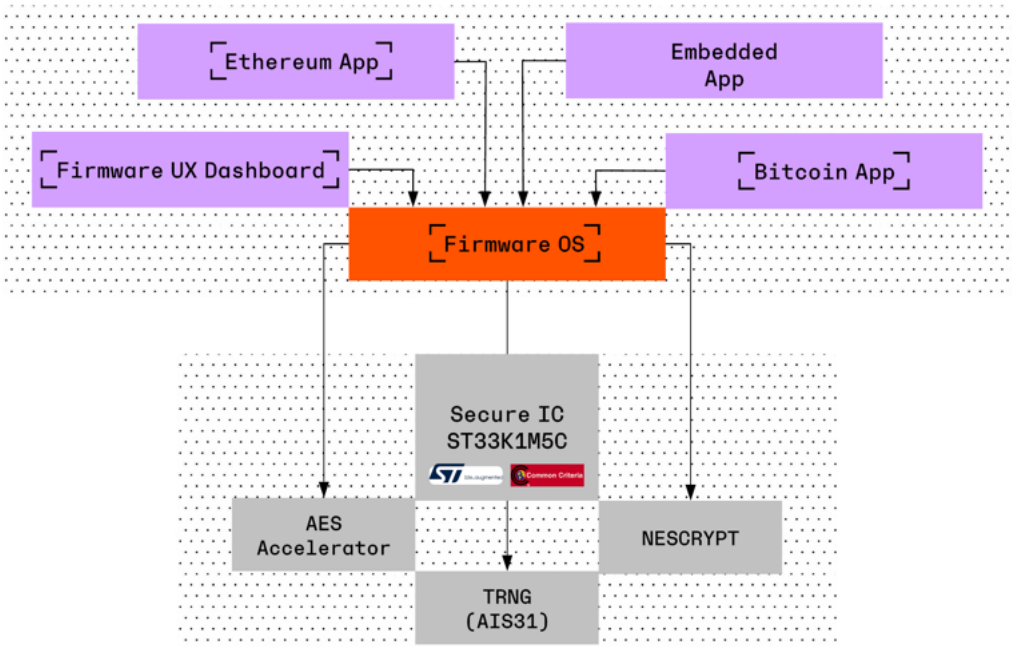

Figure 7: App Isolation on the Ledger Stax

There are two additional keys that are related to Embedded OS update process:

- The first key is NENC, which is an AES symmetric key. It is used to overencrypt the Embedded OS update over the secure channel
- The second key is UPDATE.PublicKey and is used to verify the authenticity of the OS update before installation

They are both provided during the manufacturing phase and they are stored in the device into the OS non volatile memory. They are both protected from a malicious app through the access control provided by the memory protection mechanism.

7.5.2 Application Protection

Concerning the HD path (see [Hierarchical Deterministic Wallets]) it is enforced at execution. Embedded Apps send a request to the Embedded OS to calculate the Private Key from the Secret Seed, based on a specific parameter called a derivation path.

The derivation path is unique for each crypto asset. And the Embedded App is installed with its allowed derivation paths. After having satisfied necessary security checks from the Embedded OS, the Embedded App will receive a reply with the Private Key. If the Embedded App requests a derivation path that it is not permitted to use, the request is denied. This way, many different Embedded App can be loaded onto the device, and each of them can be restricted to a specific subtree of the HD tree depending on the Embedded App's purpose.

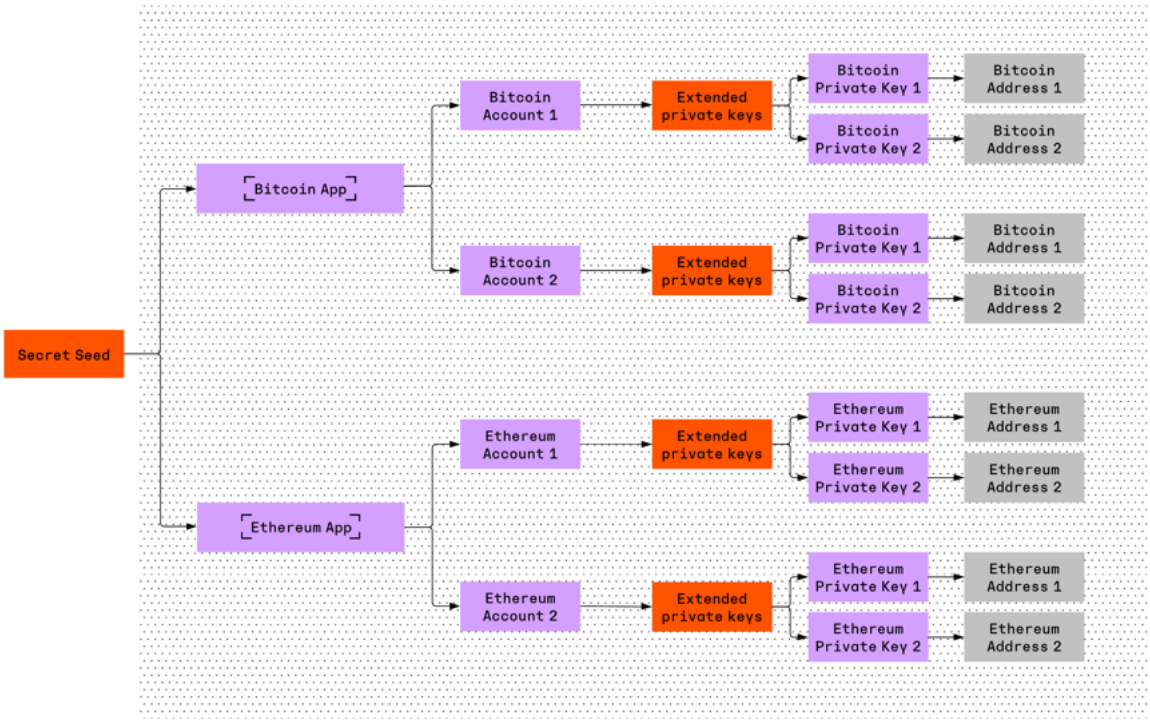

Figure 8: Key derivation from the Secret Seed

Remark: While the above paragraph is true, it only applies to Embedded Apps installed from the Ledger Live "My Ledger" section. The reason is because these Apps are verified and digitally signed by Ledger (see Security Audits and Embedded App Security).

There is another way to install Embedded Apps, called *side-loading*. This is typically used by developers (see Build & Load App). In such case the security function is not applicable anymore. Since the side-loaded Embedded App is not verified nor signed by Ledger, it can request any derivation path, as there is no way for the Embedded OS to verify the legitimacy of the request.

Therefore it is not recommended for an End-User to install on his Stax an Embedded App that is not from the Ledger Live "My Ledger" section. And if he wishes to do so he will have to perform the following actions on his device:

- As soon as you push the Embedded App on your device the End-User is presented with a "Deny unsafe manager" warning
- To pursue, the End-User has to navigate through the screen "Allow unsafe manager" and validate it
- The installation will now start and only be completed after the End-User inputs his device PIN

Furthermore, when launching a side-loaded Embedded App, a warning message indicating that the application is not authentic will appear on the screen. The user is then required to validate their action once again.

All these actions are here to avoid any malicious threat agent from side-loading an Embedded App on an End-User's device without his knowledge.

Assets:

The assets related to security function #5 are:

1. Secret Seed (Data)
2. PSD.PublicKey (Data)
3. UPDATE.PublicKey (Data)
4. NENC.PrivateKey (Data)
5. PSD.PrivateKey (Data)

8 Mapping

8.1 Mapping Between Use Cases and Security Functions

#	Use Case	SF #1	SF #2	SF #3	SF #4	SF #5
1	Checking for factory settings	-	-	-	-	-
2	Set up your Ledger Stax with Ledger Live		X			
3	Set up a PIN code			X		
4	Set up your Ledger Stax as a new device	X				
5	Restore your device from a recovery phrase	X				
6	Unlocking your device			X		
7	App installation	X	X	X	X	X
8	Adding an account				X	X
9	Sending crypto assets			X		X
10	Address verification			X		
11	Updating the Ledger Stax	X	X	X	X	
12	Updating Embedded Apps	X	X	X	X	X

The "Checking for factory settings" is covered by an assumption, as this use case can only be performed by the End-User.

8.2 Mapping Between Assets and Security Functions

#	Asset Name	SF #1	SF #2	SF #3	SF #4	SF #5
1	Random Number	-	-			
2	Secret Seed					**I & C**
3	PSD.PublicKey.Sig		**AU**			
4	PSD.PublicKey		**I**			**I**
5	PSD.Ephemeral. PrivateKey		**C**			
6	PSD.PrivateKey		**I & C**			**I & C**
7	HSM.Ephemeral. PublicKey		**I**			
8	HSM.Ephemeral. Certificate Verification		**I**			
9	HSM.PublicKey		**I & C**			
10	Secret Data protected by the PIN			**I & C**		
11	Reference PIN			**I & C**		
12	PIN Try Counter			**I**		
13	PIN Verification			**I**		
14	PIN Try Limit			**I**		
15	PIN Result			**I**		
16	Embedded OS				**I**	

#	Asset Name	SF #1	SF #2	SF #3	SF #4	SF #5
17	ECDH.Secret				**I & C**	
18	ENC.Session.key				**C**	
19	MAC.Session.key				**C**	
20	NENC.key					**AC**
21	UPDATE.PublicKey					**AC**

I = **I**ntegrity **C** = **C**onfidentiality **AU** = **AU**thenticity **AC** = **A**cces**C**ontrol

8.3 Mapping Between Security Functions and Threats

The following table gives the full relationship between the security functions and the threats.

Security Functions	Threat #1	Threat #2	Threat #3	Threat #4	Threat #5
SF #1	X				
SF #2	X	X			
SF #3			X		
SF #4		X		X	
SF #5					X

Threat #1 is also applicable to SF #2 because a nonce is required during the attestation mechanism. Besides, as SF #4 is based on SF #2, Threat #2 is also applicable to SF #4.

9 Appendix

9.1 Acronyms

Acronym	Definition
AES	**A**dvanced **E**ncryption **S**tandard
API	**A**pplication **P**rogramming **I**nterface
ANSSI	**A**gence **N**ationale de la **S**écurité des **S**ystèmes d'**I**nformation
BIP	**B**itcoin **I**mprovement **P**roposal
BSI	**B**undesamt für **S**icherheit in der **I**nformationstechnik
CC	**C**ommon **C**riteria
CSPN	**C**ertification de **S**écurité de **P**remier **N**iveau
DES	**D**ata **E**ncryption **S**tandard
EC	**E**lliptic **C**urve
ECDSA	**E**lliptic **C**urve **D**igital **S**ignature **A**lgorithm
ECDH	**E**lliptic-**C**urve **D**iffie-**H**ellman
FIDO	**F**ast **ID**entity **O**nline
HSM	**H**ardware **S**ecurity **M**odule
HTTPS	**H**yper **T**ext **T**ransfert **P**rotocol **S**ecure
IC	**I**ntegrated **C**ircuit
MCU	**M**icro **C**ontroller **U**nit
NFT	**N**on-**F**ungible **T**oken
Nonce	**N**umber used **once**
OLED	**O**rganic **L**ight **E**mitting **D**iode
PIN	**P**ersonnal **I**dentification **N**umber
PKI	**P**ublic **K**ey **I**nfrastructure
PSD	**P**ersonnal **S**ecurity **D**evice (synonym for the `Ledger Stax`)
PTC	**P**in **T**ry **C**ounter
QR code	**Q**uick **R**esponse code
RGS	**R**éférentiel **G**énéral de **S**écurité
RSA	**R**ivest **S**hamir **A**delman
SE	**S**ecure **E**lement
SEC	**S**tandards for **E**fficient **C**ryptography
SF	**S**ecurity **F**unctions
SHA	**S**ecure **H**ash **A**lgorithm
SPI	**S**erial **P**eripheral **I**nterface
ToE	**T**arget **o**f **E**valuation
TRNG	**T**rue **R**andom **N**umber **G**enerator
U2F	**U**niversal **2**nd **F**actor
UM	**U**ser **M**anual
USB	**U**niversal **S**erial **B**us
UX	**U**ser e**X**perience

9.2 Terminology

Terminology	Definition
Adversary	Person trying to compromise the `Ledger Stax`.

Terminology	Definition
Attestation	One of the core security features developed by LEDGER to prove by cryptographic means the Ledger Stax is genuine. The attestation mechanism implementation relies on a set of cryptographic protocols based on Elliptic Curve.
Blockchain	A list of blocks which are all linked together and validated via a consensus mechanism.
Command/Response	The Host and the Ledger Stax exchanges though a set of commands/responses (e.g. VALIDATE_TARGET_ID, INITIALIZE_AUTHENTICATION, VALIDATE_CERTIFICATE_LAST).
Consent	The Ledger Stax security design is strengthened by the End-User. As soon as a sensitive operation is required, the End-User must confirm the operation via the touch screen.
Crypto Asset	One of the digital asset whose value is saved on the blockchain.
Crypto Asset address	It is a public address provided by the End-User to transfer crypto assets.This address is derived from the Public Key.
E-ink (electronic ink)	Is a display device that mimics the appearance of ordinary ink on paper.
Embedded App	Software running in the SE on top of the Embedded OS. These Embedded Apps can be either developed by LEDGER or a third-party. A Embedded App offers a service.
Embedded OS	The open native Operating System developed by LEDGER running on top of the secure IC. One of it's features is to manage Apps (delete, install) after issuance on the field. This capability offering flexibility allows to enrich the Ledger Stax experience.
End-User	Owner of a Ledger Stax. End-User is defined by general public.
Firmware	Software running on top of the MCU hardware.
Hardware Wallet	Physical wallet leveraging hardware to secure sensitive assets and sensitive operations.
Host	End-User machine (desktop, laptop, tablet or smartphone) running [Ledger Live].
Key Pair	Includes both a Private Key and a Public Key.
Ledger Live	[Ledger Live] a companion app running on the Host to support the Ledger Stax services.
Ledger Live Mobile	Smartphone version of [Ledger Live] for Android and iOS smartphones.
Ledger Stax	State-of-the-art device designed, developed and manufactured by LEDGER offering a set of secure services. In this Security Target, PSD and Ledger Stax are interchangeable.
NESCRYPT	Coprocessor for public key cryptography algorithm embedded in [ST33K1M5C]. For instance, LEDGER leverages NESCRYPT to perform some operations on the elliptic curve.
Onboarding	Set of operations (seed generation, PIN configuration...) performed during the initialization of the Ledger Stax.
Private Key	Set of secret data involved for signing a transaction under the End-User Control.

Terminology	Definition
Public Key	Set of data, generated from the private key, distributed and used to verify the signature.
secp256k1	Elliptic Curve defined by Certicom Research in Standards for Efficient Cryptography ([SEC_2]).
Secure Element	A Secure Element is composed of a secure IC and a Secure Software.
Secure IC	It is a hardware embedding a set of physical security countermeasures. The Secure IC included in the Ledger Stax is Common Criteria certified [ST33_CC].
Secure Software	It is a software embedding a set of logical security countermeasures. In the Ledger Stax, LEDGER has developed Embedded OS and a set of Embedded Apps for the Ledger Stax.
Seed	A set of data located at the top of a hierarchical tree. In the LEDGER context it refers to the master key which every application on a Ledger device uses to calculate their private keys from.
Service	Crypto asset management, Password Manager, Second Factor Authentication are typical services offered by the Ledger Stax.
Wallet	Solution to manage your crypto assets (hardware wallet, software wallet, paper wallet...).
Wallet Type	There are 2 types of wallets: non-deterministic wallet and deterministic wallet.