BOLOS Python Loader

Release 0.1.15

Jul 13, 2018

Contents

The BOLOS Python loader is a Python library and collection of scripts for interfacing with and managing BOLOS devices from a host computer. See the Python loader GitHub repository for download and installation instructions.

CHAPTER 1

Scripts

The Python loader includes a collection of useful scripts for managing BOLOS devices. This section includes an overview of some of the most important scripts and how they can be used.

In order to use any of these scripts, the device must be in the dashboard application (no apps are open, the device should display a list of installed apps).

Here is an example using the *deleteApp.py* script from the command-line:

```
python -m ledgerblue.deleteApp --targetId 0x31100002 --appName "Hello World"
```

The above command will delete the app named "Hello World" from the connected Leger Nano S.

See the *Script Reference* for the detailed documentation about each script.

CHAPTER 2

Script Reference

2.1 checkGenuine.py

Use attestation to determine if the device is a genuine Ledger device.

```
usage: python -m ledgerblue.checkGenuine [-h] [--targetId TARGETID]
                                         [--issuerKey ISSUERKEY] [--apdu]
```

2.1.1 Named Arguments

--targetId The device's target ID (default is Ledger Blue)

--issuerKey Issuer key (hex encoded, default is batch 1)

--apdu Display APDU log

Default: False

2.2 deleteApp.py

Delete the app with the specified name.

```
usage: python -m ledgerblue.deleteApp [-h] [--targetId TARGETID]
                                      [--appName APPNAME] [--appHash APPHASH]
                                      [--rootPrivateKey ROOTPRIVATEKEY]
                                      [--apdu] [--deployLegacy]
```

2.2.1 Named Arguments

--targetId The device's target ID (default is Ledger Blue)

--appName	The name of the application to delete
--appHash	Set the application hash
--rootPrivateKey	A private key used to establish a Secure Channel (hex encoded)
--apdu	Display APDU log Default: False
--deployLegacy	Use legacy deployment API Default: False

2.3 derivePassphrase.py

Set a BIP 39 passphrase on the device.

```
usage: python -m ledgerblue.derivePassphrase [-h] [--persistent]
```

2.3.1 Named Arguments

--persistent	Persist passphrase as secondary PIN (otherwise, it's set as a temporary passphrase) Default: False

2.4 endorsementSetupLedger.py

Generate an attestation keypair, using Ledger to sign the Owner certificate.

```
usage: python -m ledgerblue.endorsementSetupLedger [-h] [--url URL] [--apdu]
                                                   [--perso PERSO]
                                                   [--endorsement ENDORSEMENT]
                                                   [--targetId TARGETID]
                                                   [--key KEY]
```

2.4.1 Named Arguments

--url	Server URL Default: "https://hsmprod.hardwarewallet.com/hsm/process"
--apdu	Display APDU log Default: False
--perso	A reference to the personalization key; this is a reference to the specific Issuer keypair used by Ledger to sign the device's Issuer Certificate Default: "perso_11"
--endorsement	A reference to the endorsement key to use; this is a reference to the specific Owner keypair to be used by Ledger to sign the Owner Certificate Default: "attest_1"

--targetId	The device's target ID (default is Ledger Blue)
--key	Which endorsement scheme to use (1 or 2)

2.5 endorsementSetup.py

Generate an attestation keypair, using the provided Owner private key to sign the Owner Certificate.

```
usage: python -m ledgerblue.endorsementSetup [-h] [--key KEY]
                                             [--certificate CERTIFICATE]
                                             [--privateKey PRIVATEKEY]
                                             [--targetId TARGETID]
                                             [--issuerKey ISSUERKEY] [--apdu]
```

2.5.1 Named Arguments

--key	Which endorsement scheme to use (1 or 2)
--certificate	Optional certificate to store if finalizing the endorsement (hex encoded), if no private key is specified
--privateKey	Optional private key to use to create a test certificate (hex encoded), if no certificate is specified
--targetId	The device's target ID (default is Ledger Blue)
--issuerKey	Issuer key (hex encoded, default is batch 1)
--apdu	Display APDU log Default: False

2.6 genCAPair.py

Generate a Custom CA public-private keypair and print it to console.

```
usage: python -m ledgerblue.genCAPair [-h]
```

2.7 hashApp.py

Calculate an application hash from the application's hex file.

```
usage: python -m ledgerblue.hashApp [-h] [--hex HEX]
```

2.7.1 Named Arguments

--hex	The application hex file to be hashed

2.8 hostOnboard.py

Warning: Using this script undermines the security of the device. Caveat emptor.

```
usage: python -m ledgerblue.hostOnboard [-h] [--apdu] [--id ID] [--pin PIN]
                                        [--prefix PREFIX]
                                        [--passphrase PASSPHRASE]
                                        [--words WORDS]
```

2.8.1 Named Arguments

--apdu	Display APDU log
	Default: False
--id	Identity to initialize
--pin	Set a PINs to backup the seed for future use
--prefix	Derivation prefix
--passphrase	Derivation passphrase
--words	Derivation phrase

2.9 listApps.py

List all apps on the device.

```
usage: python -m ledgerblue.listApps [-h] [--targetId TARGETID]
                                     [--rootPrivateKey ROOTPRIVATEKEY]
                                     [--apdu] [--deployLegacy]
```

2.9.1 Named Arguments

--targetId	The device's target ID (default is Ledger Blue)
--rootPrivateKey	The Signer private key used to establish a Secure Channel (otherwise, a random one will be generated)
--apdu	Display APDU log
	Default: False
--deployLegacy	Use legacy deployment API
	Default: False

2.10 loadApp.py

Load an app onto the device from a hex file.

```
usage: python -m ledgerblue.loadApp [-h] [--targetId TARGETID]
                                    [--fileName FILENAME] [--icon ICON]
                                    [--curve CURVE] [--path PATH]
                                    [--appName APPNAME]
                                    [--signature SIGNATURE] [--signApp]
                                    [--appFlags APPFLAGS]
                                    [--bootAddr BOOTADDR]
                                    [--rootPrivateKey ROOTPRIVATEKEY]
                                    [--signPrivateKey SIGNPRIVATEKEY] [--apdu]
                                    [--deployLegacy] [--apilevel APILEVEL]
                                    [--delete] [--params] [--tlv]
                                    [--dataSize DATASIZE]
                                    [--appVersion APPVERSION] [--offline]
                                    [--installparamsSize INSTALLPARAMSSIZE]
                                    [--tlvraw TLVRAW] [--dep DEP]
```

2.10.1 Named Arguments

--targetId	The device's target ID (default is Ledger Blue)
--fileName	The application hex file to be loaded onto the device
--icon	The icon content to use (hex encoded)
--curve	A curve on which BIP 32 derivation is locked ("secp256k1", "prime256r1", or "ed25519"), can be repeated
--path	A BIP 32 path to which derivation is locked (format decimal a'/b'/c), can be repeated
--appName	The name to give the application after loading it
--signature	A signature of the application (hex encoded)
--signApp	Sign application with provided signPrivateKey
	Default: False
--appFlags	The application flags
--bootAddr	The application's boot address
--rootPrivateKey	The Signer private key used to establish a Secure Channel (otherwise a random one will be generated)
--signPrivateKey	Set the private key used to sign the loaded app
--apdu	Display APDU log
	Default: False
--deployLegacy	Use legacy deployment API
	Default: False
--apilevel	Use given API level when interacting with the device
--delete	Delete the app with the same name before loading the provided one
	Default: False
--params	Store icon and install parameters in a parameter section before the code
	Default: False

--tlv	Use install parameters for all variable length parameters Default: False
--dataSize	The code section's size in the provided hex file (to separate data from code, if not provided the whole allocated NVRAM section for the application will remain readonly.
--appVersion	The application version (as a string)
--offline	Request to only output application load APDUs Default: False
--installparamsSize	The loaded install parameters section size (when parameters are already included within the .hex file.
--tlvraw	Add a custom install param with the hextag:hexvalue encoding
--dep	Add a dependency over an appname[:appversion]

2.11 loadMCU.py

Load the firmware onto the MCU. The MCU must already be in bootloader mode.

```
usage: python -m ledgerblue.loadMCU [-h] [--targetId TARGETID]
                                    [--fileName FILENAME]
                                    [--bootAddr BOOTADDR] [--apdu] [--reverse]
                                    [--nocrc]
```

2.11.1 Named Arguments

--targetId	The device's target ID
--fileName	The name of the firmware file to load
--bootAddr	The firmware's boot address
--apdu	Display APDU log Default: False
--reverse	Load HEX file in reverse from the highest address to the lowest Default: False
--nocrc	Load HEX file without checking CRC of loaded sections Default: False

2.12 mcuBootloader.py

Request the MCU to execute its bootloader.

```
usage: python -m ledgerblue.mcuBootloader [-h] [--targetId TARGETID]
                                          [--rootPrivateKey ROOTPRIVATEKEY]
                                          [--apdu]
```

2.12.1 Named Arguments

--targetId The device's target ID (default is Ledger Blue)

--rootPrivateKey The Signer private key used to establish a Secure Channel (otherwise a random one will be generated)

--apdu Display APDU log

Default: False

2.13 resetCustomCA.py

Remove all Custom CA public keys previously enrolled onto the device.

```
usage: python -m ledgerblue.resetCustomCA [-h] [--targetId TARGETID] [--apdu]
                                          [--rootPrivateKey ROOTPRIVATEKEY]
```

2.13.1 Named Arguments

--targetId The device's target ID (default is Ledger Blue)

--apdu Display APDU log

Default: False

--rootPrivateKey The Signer private key used to establish a Secure Channel (otherwise a random one will be generated)

2.14 runApp.py

```
usage: python -m ledgerblue.runApp [-h] [--targetId TARGETID] [--apdu]
                                   [--rootPrivateKey ROOTPRIVATEKEY]
                                   [--appName APPNAME]
```

2.14.1 Named Arguments

--targetId The device's target ID (default is Ledger Blue)

--apdu Display APDU log

Default: False

--rootPrivateKey The Signer private key used to establish a Secure Channel (otherwise a random one will be generated)

--appName The name of the application to run

2.15 runScript.py

Read a sequence of command APDUs from a file and send them to the device. The file must be formatted as hex, with one CAPDU per line.

```
usage: python -m ledgerblue.runScript [-h] [--fileName FILENAME] [--apdu]
                                      [--scp] [--targetId TARGETID]
                                      [--rootPrivateKey ROOTPRIVATEKEY]
```

2.15.1 Named Arguments

--fileName The name of the APDU script to load

--apdu Display APDU log

Default: False

--scp Open a Secure Channel to exchange APDU

Default: False

--targetId The device's target ID (default is Ledger Nano S)

--rootPrivateKey The Signer private key used to establish a Secure Channel (otherwise a random one will be generated)

2.16 setupCustomCA.py

Enroll a Custom CA public key onto the device.

```
usage: python -m ledgerblue.setupCustomCA [-h] [--targetId TARGETID] [--apdu]
                                          [--rootPrivateKey ROOTPRIVATEKEY]
                                          [--public PUBLIC] [--name NAME]
```

2.16.1 Named Arguments

--targetId The device's target ID (default is Ledger Blue)

--apdu Display APDU log

Default: False

--rootPrivateKey The Signer private key used to establish a Secure Channel (otherwise a random one will be generated)

--public The Custom CA public key to be enrolled (hex encoded)

--name The name to assign to the Custom CA (this will be displayed on screen upon auth requests)

2.17 signApp.py

2.18 updateFirmware.py

```
usage: python -m ledgerblue.updateFirmware [-h] [--url URL] [--apdu]
                                           [--perso PERSO]
                                           [--firmware FIRMWARE]
                                           [--targetId TARGETID]
                                           [--firmwareKey FIRMWAREKEY]
```

2.18.1 Named Arguments

--url Server URL

Default: "https://hsmprod.hardwarewallet.com/hsm/process"

--apdu Display APDU log

Default: False

--perso A reference to the personalization key; this is a reference to the specific Issuer keypair used by Ledger to sign the device's Issuer Certificate

Default: "perso_11"

--firmware A reference to the firmware to load

--targetId The device's target ID (default is Ledger Blue)

--firmwareKey A reference to the firmware key to use

2.19 verifyApp.py

```
usage: python -m ledgerblue.verifyApp [-h] [--hex HEX] [--key KEY]
                                      [--signature SIGNATURE]
```

2.19.1 Named Arguments

--hex The hex file of the signed application

--key The Custom CA public key with which to verify the signature (hex encoded)

--signature The signature to be verified (hex encoded)

2.20 verifyEndorsement1.py

Verify a message signature created with Endorsement Scheme #1.

```
usage: python -m ledgerblue.verifyEndorsement1 [-h] [--key KEY]
                                               [--codehash CODEHASH]
                                               [--message MESSAGE]
                                               [--signature SIGNATURE]
```

2.20.1 Named Arguments

--key	The endorsement public key with which to verify the signature (hex encoded)
--codehash	The hash of the app associated with the endorsement request (hex encoded)
--message	The message associated to the endorsement request (hex encoded)
--signature	The signature to be verified (hex encoded)

2.21 verifyEndorsement2.py

Verify a message signature created with Endorsement Scheme #2.

```
usage: python -m ledgerblue.verifyEndorsement2 [-h] [--key KEY]
                                               [--codehash CODEHASH]
                                               [--message MESSAGE]
                                               [--signature SIGNATURE]
```

2.21.1 Named Arguments

--key	The endorsement public key with which to verify the signature (hex encoded)
--codehash	The hash of the app associated with the endorsement request (hex encoded)
--message	The message associated to the endorsement request (hex encoded)
--signature	The signature to be verified (hex encoded)