MemType Documentation

area0x33

Table of Contents

This documentation is under active development and discussion!

If you find errors or omissions in this document, please don’t hesitate to submit an issue or open a pull request with a fix. We also encourage you to ask questions and discuss any aspects of the project by email.

Figure 1. MemType

Welcome to the MemType documentation, here you’ll learn everything about the MemType hardware, firmware, software tools and usage for any user level.

If you fancy one, you can get it from our tindie store!

Introduction

Figure 2. MemType top face

Technically speaking the MemType is an open source `hardware password keeper`. This means it is basically a device focused in securely `storing passwords`. The `open source` part means that we share all the software source files, hardware design files and documentation under the License conditions.

Having the passwords stored in a password keeper has several advantages:

- No reusing same pass in different apps or webs.
- More secure and long passwords can be used, no need to remember them.
- No more forgotten passwords.

We like hardware password keepers but we couldn't find one that satisfied our usability needs, also they were very expensive! That's why we decided to make the MemType `loaded with features` and with a `reduced BOM` that results in a simple `low cost` device that's also great for `DIY makers` and `hackers`.

MemType features

- `Encrypted storage` in the device using NOEKEON written in pure assembler!
- `AES encrypted backups` on the PC using the MemTypeTool GUI.
- Plug it into a `USB`, `no drivers or software for using it`. [1: Using it doesn't require software or drivers. For managing operations like storing credentials or changing the PIN the MemTypeTool is needed and depending on the OS the libusb driver.]! Works as a HID keyboard.
- `Works anywhere!`. While others only work on webs or certain apps, the MemType can be used anywhere from webs, apps, terminals or anywhere it's allowed to type using a keyboard.
- `One click login`. It stores and `types for you` not only the `password` but also the `user` without any interaction from your part, and it also submits it!
- Add `delays` to `enter the password when it's asked`, for example in `ssh` or login forms where the user and password are asked in two steps.
- `Dynamic storage`. You're `not limited` to a certain number of credentials, just fill the 2KB storage with them.
- `Assign names` to the credentials so you can find them easy and fast to use them.
- Manage the device with the MemTypeTool GUI software, super easy and simple!
- `Truly multiplatform`. The MemType works on `any device with usb` keyboard capability like tablets, smart tv, and phones.
- `Any keyboard layout`. Load the keyboard layout you've got working in your OS using the MemTypeTool GUI.
- `Truly Open Source` Software and Hardware.

Credentials

The data structure used by the MemType to store `passwords` and `usernames` is called a `credential`. Appart from passwords and usernames it also stores a field that stores the key combination to move from the user input to the password input, called `hop`. It also includes a field for storing the key combination to submit the login form or command, called `submit`. To help you find the credential you need, each of them has a name stored in the `name` field.

Field	Description
name	The name that'll be shown while using the device.
user	The username of the login.
hop	The sequence of keyboard keys to move from the user input field to the password one. Usually `tab`, sometimes `Enter`
password	The password that'll be typed into the password input field.
submit	The sequence of keyboard keys to validate the login. Usually `Enter`.

When a `credential` is applied, the MemType generates the key sequence like you were typing it into your keyboard, excluding the name by the order of the table above.

In a login form, you focus the username field and then move the MemType joystick `up` and `down` looking for the credential you need. When you find the right credential, `enter` to apply it and the MemType will erase the credential name, type the username, move to the password input field, type it and submit. All this process is done much times faster than any person can type.

There are some cases where you may need a `delay` in a certain point, for example when the login form asks for the user, validates it and then asks for the password. Such delays can be made by the MemType using one special character when editing the credential. This and more is explained in the Advanced uses section.

It's easy to see how this credential structure can be used to `securely store` and type other kind of data like commands or `credit card numbers`.

When buying online it's faster and more secure to let the MemType type the credit card number than pulling out the creditcard with the number printed on it.

A way of accomplishing it would be by having all the fields empty except the `password` field that would contain the `credit card number` and the name.

Links

N_O_D_E Video review

If you want a great video introduction and review of the MemType, check the following awesome video made by N_O_D_E.

▶ https://www.youtube.com/watch?v=O16U_TjAutU *(YouTube video)*

Compare to others password keepers

Quick start

Open a `text editor` and connect the MemType to the `USB` port of your computer. You'll see the `red led` ● turning on indicating it's being `powered` correctly and it's `locked`.

On the text editor it will write a `welcome message` indicating it's firmware version:

```
MemType 3.1.0
```

You don't have to delete what it writes, it deletes it's own text to write new one, `that's how the MemType communicates with you`. Now you can unlock it entering the `PIN`, by default it's `0000`. To enter the PIN use the `joystick`:

- up ↑ increases the number.
- down ↓ decreases.
- towards the connector → accepts the number.
- the opposite joystick movement ← to cancel.

We'll refer to these movements `up`, `down`, `enter` and `back` respectively.

You should have now the pin entered:

```
PIN: 0000
```

One more `enter` and it'll validate and show the name of the first credential and the green led ● will be on. If the PIN is incorrect an error message will be displayed:

```
PIN ERR
```

The MemType comes preloaded with 3 dummy credentials so you can play the first time. Move `up` and `down` to see their names, apply one with the `enter` movement on the joystick, see the details with one `back` movement and lock the device with one more `back` movement (2 in total) turning the red led ● on again.

That's the basic usage of the MemType, now to be useful it has to contain the credentials you'll use, setup a **different PIN**, and be loaded with the same `keyboard layout` as your operating system or you'll see strange characters. To do all this management on the device there is an easy to use Graphical User Interface software called MemTypeTool GUI.

MemTypeTool GUI

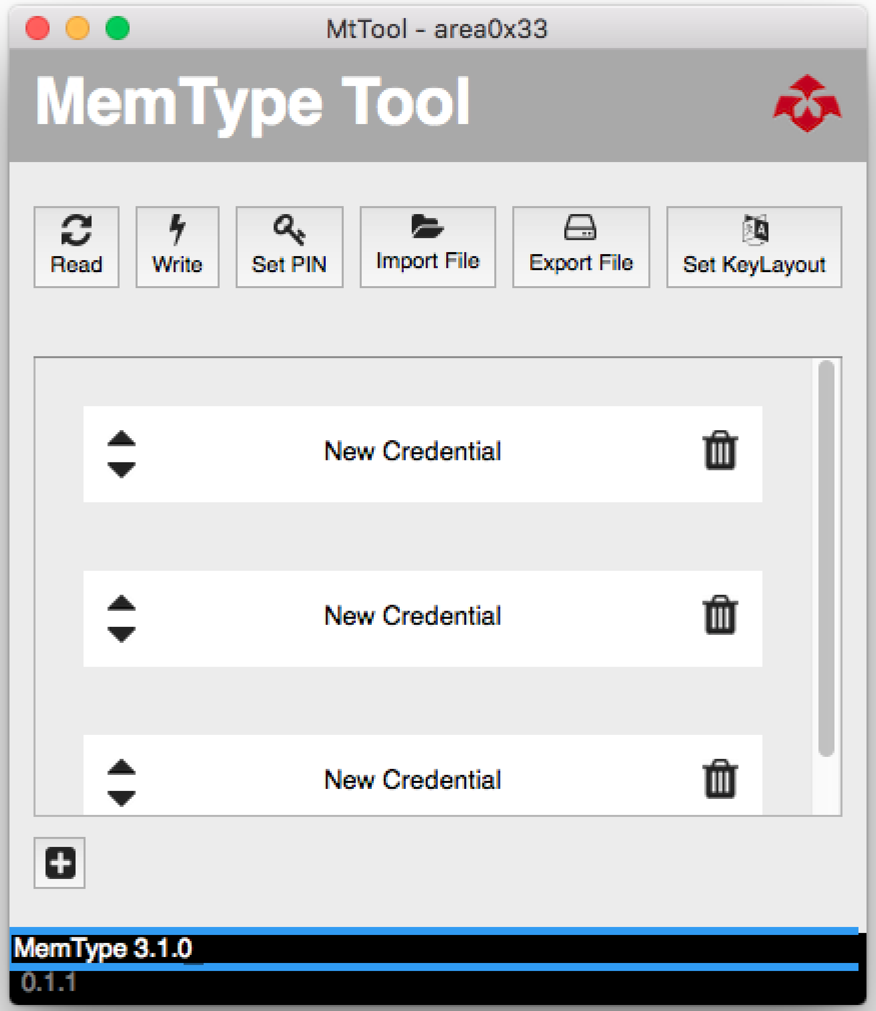


Figure 3. MemTypeTool GUI

The MemTypeTool GUI can be downloaded from : https://github.com/oyzzo/MemTypeTool

It is developed using python 2.7 , pyqt4, and libusb so it should work anywhere you can get this dependencies running. It has been tested on Windows, Linux and MacOS.

To run the `MemTypeTool`:

```
python main.py
```

Windows installation

It has been tested in windows up to 8.1. It has been reported to not work in windows 10.

Using a linux virtual machine can be a workaround for the moment.

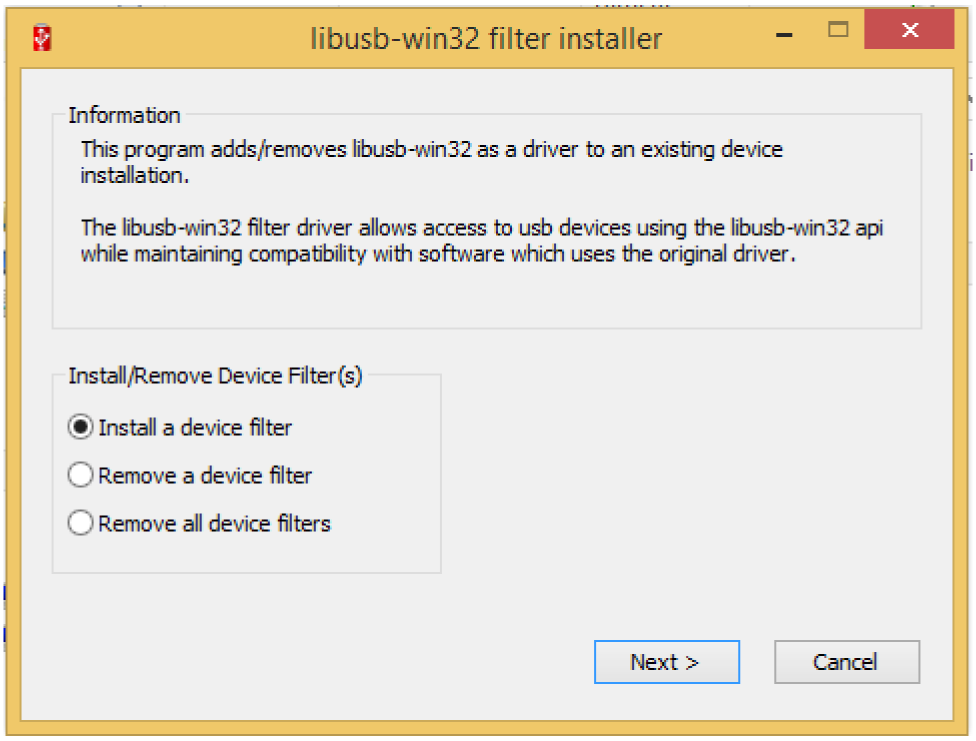
1. Install python 2.7.
    a. https://www.python.org/downloads/
2. Add python to your path:

```
C:\Python27\;C:\Python27\Scripts\
```

3. Download and install PyQt for python 2.7.
    a. https://sourceforge.net/projects/pyqt/files/PyQt4/PyQt-4.11.4/
4. Download, unzip and install libusb:
    a. https://github.com/walac/pyusb/archive/master.zip

    ```
    python setup.py install
    ```

5. Plug in the Memtype and let windows install the drivers for the device (we'll change them in the next step).
6. Download and extract libusb-win32:
    a. http://sourceforge.net/projects/libusb-win32/files/libusb-win32-releases/1.2.6.0/libusb-win32-bin-1.2.6.0.zip/download
7. Execute install-filter-win (find it inside extracted folder, under /bin/x86 or `other if your windows is 64bits`).
    a. Select Install Filter, next.


libusb-win32 filter installer

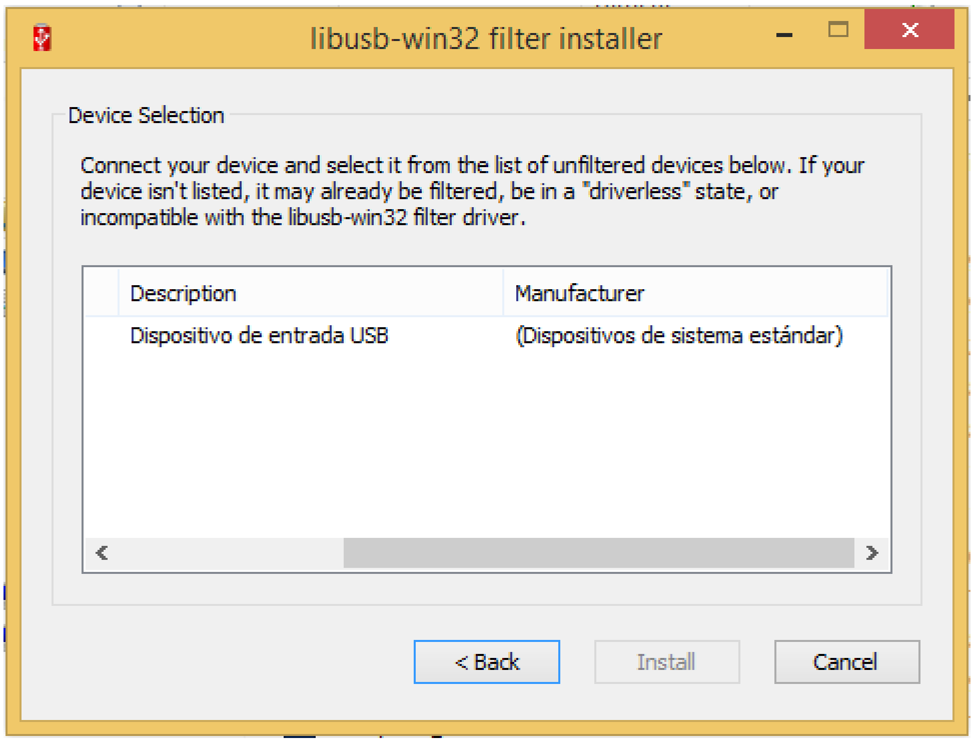
Information
This program adds/removes libusb-win32 as a driver to an existing device installation.

The libusb-win32 filter driver allows access to usb devices using the libusb-win32 api while maintaining compatibility with software which uses the original driver.

Install/Remove Device Filter(s)
- [✓ Install a device filter]
- Remove a device filter
- Remove all device filters

Figure 4. Windows libusb filter, step 1

a. Select Memtype (pid a033) and click Install

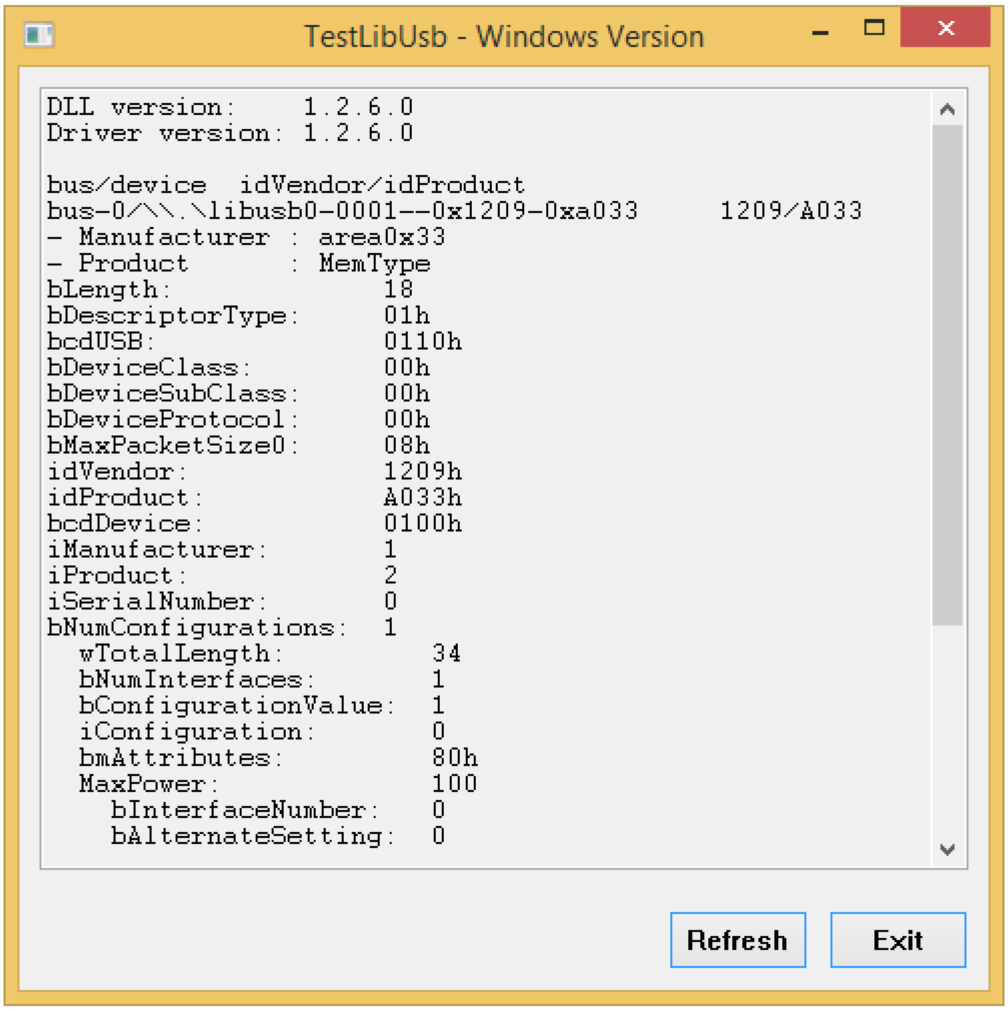


Connect your device and select it from the list of unfiltered devices below. If your device isn't listed, it may already be filtered, be in a "driverless" state, or incompatible with the libusb-win32 filter driver.

Description	Manufacturer
Dispositivo de entrada USB	(Dispositivos de sistema estándar)

Figure 5. Windows libusb filter, step 2

a. Done!!! you can check it with the testlibusb-win found next to install-filter-win:



```
DLL version:     1.2.6.0
Driver version:  1.2.6.0

bus/device  idVendor/idProduct
bus-0/\\.\libusb0-0001--0x1209-0xa033     1209/A033
- Manufacturer : area0x33
- Product      : MemType
bLength:              18
bDescriptorType:      01h
bcdUSB:               0110h
bDeviceClass:         00h
bDeviceSubClass:      00h
bDeviceProtocol:      00h
bMaxPacketSize0:      08h
idVendor:             1209h
idProduct:            A033h
bcdDevice:            0100h
iManufacturer:        1
iProduct:             2
iSerialNumber:        0
bNumConfigurations:   1
  wTotalLength:          34
  bNumInterfaces:        1
  bConfigurationValue:   1
  iConfiguration:        0
  bmAttributes:          80h
  MaxPower:              100
    bInterfaceNumber:    0
    bAlternateSetting:   0
```

Figure 6. Check windows filter

Linux installation

Example ubuntu installation steps:

```
apt-get install python
apt-get install python-pyqt4
pip install pyusb
```

MacOS installation

Usage

You can use the `bottom input text` to communicate with your memtype, `focusing it` with a mouse click if it's not already focused.

▶ unlock.mp4 *(video)*

Setting the keyboard layout

Using the GUI you can change the `MemType keyboard layout` to match your `computer keyboard layout`. The current available layouts files are located under `keyboard directory`.

▶ setkeyboard.mp4 *(video)*

Setting the P.I.N.

`Changing the default PIN` is a very important action that `should be done as soon as possible` before loading any credential into de MemType.

▶ setpin.mp4 *(video)*

Reading from device

You can `read the credentials from the MemType` to do any action to them and write them back, or make an `encrypted` file `backup` by Exporting encrypted file.

▶ read.mp4 *(video)*

Writting to device

When you've got everything looking OK in the MemTypeTool GUI you have to `write the credentials` to the MemType device so you can use them.

Another option would be to simply store them by Exporting encrypted file.

▶ write.mp4 *(video)*

Adding credentials

Adding credentials is easy, when you've got your new credentials created remember Writting to device.

▶ add.mp4 *(video)*

Editing credentials

Editing is essential, edit to have your passwords in the credentials, usernames and any other field so you can forget them!

▶ edit.mp4 *(video)*

Deleting credentials

Delete by clicking into the `bin icon` of any credential. The `changes` will not take effect until you `write` to the device.

▶ delete.mp4 *(video)*

Exporting encrypted file

You can have backups in your computer, or have one AES encrypted file containing all the credentials for a given user role in your company.

⚠ Don't forget the file password or there'll be no way of getting back the credentials from that file.

▶ export.mp4 *(video)*

Importing from encrypted file

As easy as exporting, enter the file password and your credentials will be ready to be edited.

▶ import.mp4 *(video)*

Adding delays to credentials

Using the MemType

P.I.N. Unlock

Use a credential

Credential details

Lock the device

Advanced uses

MemType library

The MemType library (`libsmttool`) is a `python library` that allows `communication` with the MemType in python. It allows all the operations that can be realized with the MemTypeTool GUI as it's based on libsmttool.

It can be found in this github: https://github.com/oyzzo/MemTypeTool/tree/master/libsmttool.

It also can be found under the `libsmttool` directory when downloading the MemTypeTool GUI. https://github.com/oyzzo/MemTypeTool/archive/master.zip

It's composed of two modules, the `noekeon.py` file and the `memtype.py` file.

This library depends on:

- python2.7
- libusb-1.0-0
- pyusb

noekeon.py

As the memtype uses internally the noekeon algorithm for encrypting the credentials, this algorithm has to be used in the MemTypeTool GUI to decrypt them back.

This file provides a python implementation based on the noekeon specification.

NOEKEON is an iterated block cipher with a block and key length of 128 bits.

The public methods implemented in the library are:

1. **NoekeonEncrypt(WorkingKey, State)**
2. **NoekeonDecrypt(WorkingKey, State)**

The Number of Rounds is defined by `NUMBER_OF_ROUNDS` and fixed to `16`.

WorkingKey format is 4 elements of 32 bit length each.

State is the information to be encrypted / decrypted in `4 elements of 32 bit`.

memtype.py

MemType command library implementation.

Public methods implemented in `memtype.py` file:

1. **memtype Class** Communicates with the device.
    a. **connect** Open USB connection with device.
    b. **disconnect** Close USB connection with device.
    c. **info** Read info from device and return deviceInfo object.
    d. **write** Write credentials to memtype credential reserved memory area, credential data must be encrypted with the right PIN.
    e. **read** Read credentials from memtype, must be decrypted with the right PIN.
    f. **writePinHash** Writes PIN hash into memtype, **pinToHas** function can be used to generate the hash.
    g. **readPinHash** Reads the PIN hash from the memtype.
    h. **writeKeyboardLayout** Writes a keyboard layout into the memtype device.
    i. **isLocked** Returns TRUE if the memtype has not been unlocked with PIN.
    j. **validatePin** When the memtype is unlocked, checks if a PIN corresponds to the PIN hash and returns TRUE, FALSE otherwise.
2. **decryptCredentialList(cl, key)** Decrypts a credential list according to device implementation.
3. **encryptCredentialList(cl, key)** Encrypts a credential list according to device implementation.

Hardware

PCB

BOM

Firmware

License

F.A.Q.

Can I make a MemType myself?

Yes! It’s a great DIY project!. You can make your PCB using the design files or order one from OSHpark, buy the components, solder them, compile the firmware and burn it into the microcontroller!

How many credentials can be stored?

The capacity of the credentials storage on the MemType is `2KB`. Credentials are stored dinammically so `it’ll depend on the length` of the name, username, password etc…

Can I make a backup of my credentials?

Having a backup of the credentials is always a good idea. You can export an encrypted file from the `MemTypeGUI` containning all your credentials.