MemType Getting Started Guide

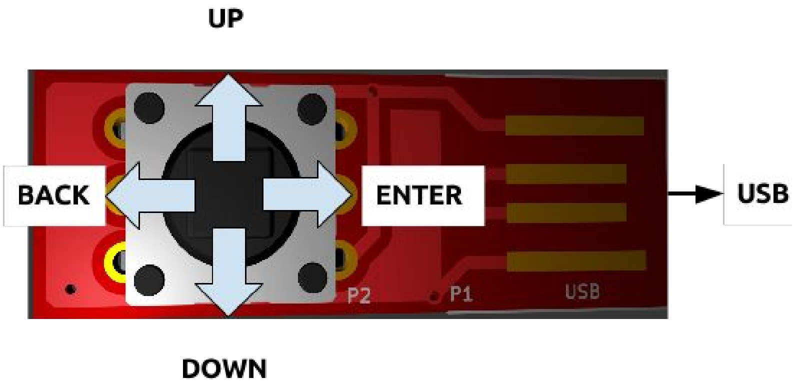
www.area0x33.com

1.- Memtype Overview

The memtype is a simple device that can store a set of encrypted username and passwords using a credential structure. It can also type them for you so you don’t have to, type the tab to move from user to password fields or click enter to submit and more.

To manage device credentials there is a user command line tool called “mttool” that allows to read and write credentials to and from the computer, change the pin and load ascii to keyboard configuration files into device.

1.1.- Joystick actions:

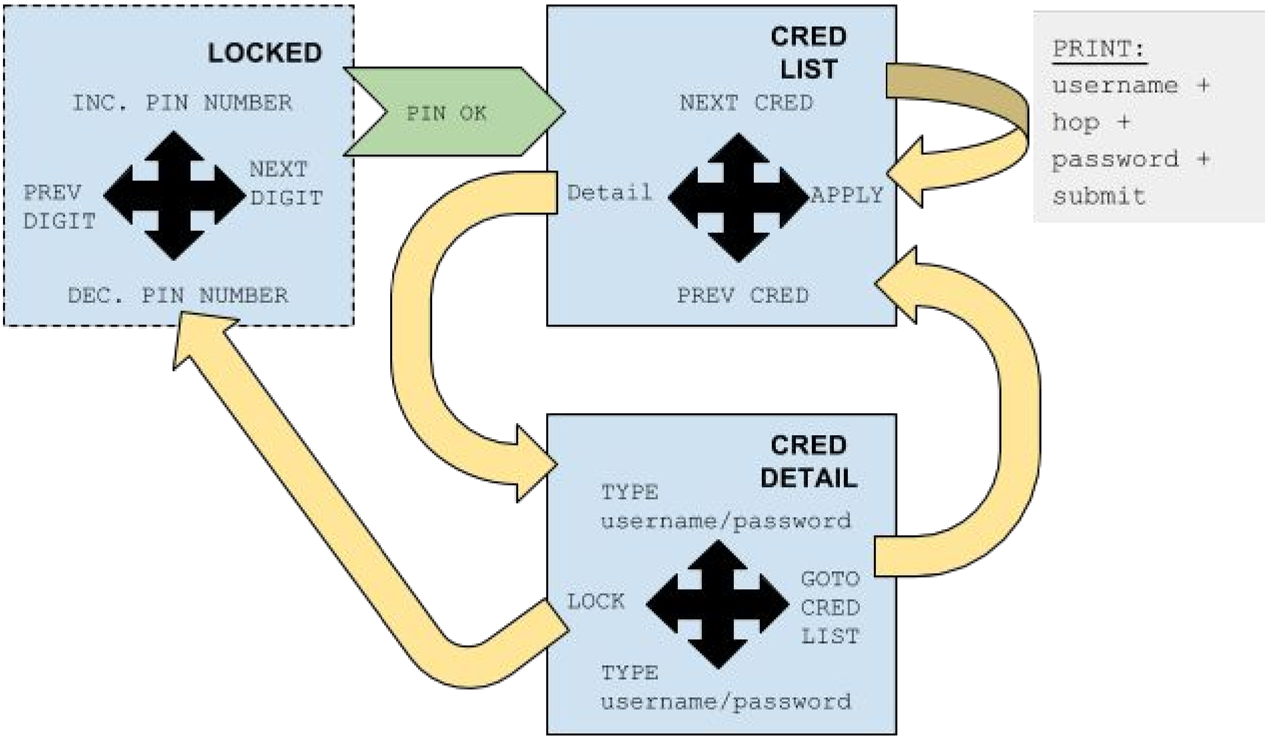


1.2.- Credential structure:

A memtype credential is a basic structure formed by 5 strings:

name: A name to identify the credential on the list.
username: The actual username of the service/web/application etc.
hop: A key combination to move from the user input to the password input (usually a tab).
password: The secret and secure password corresponding to the username.
submit: A key combination to apply the log-in (usually an enter).

1.3.- Memtype States:



2.- Connect your memtype

So open and focus on a terminal, text editor or the url bar of the browser, any place where you can type things will do. That will allow the memtype to type messages for you to read. The next thing you have to do is to plug in the memtype to a usb port on your computer. It'll print a message similar to this one:
“`MemType 3.0.0`”

3.- Unlock the memtype

The default state of the memtype is locked. To unlock it you have to enter the PIN. The default PIN is “0000” but you can change it using the command line tool.

In order to enter the PIN numbers you have to move the joystick. Joysticks actions are shown on image 1.

Moving it up will increase the number, moving it down will decrease it and moving it towards the usb port will enter the number.

So doing an enter will print “PIN: 0” do it 3 more times and when it prints “`PIN: 0000`” enter it with the same joystick movement again.

When it’s unlocked It’ll enter to the credential list printing the first one.

4.- Credential list

The main state of the memtype device is the credential list and it’s a simple list of credential names. Moving up or down will change the name of the credential to the preceding or following respectively.

Two actions can be done the the shown credential, the first and more useful is to enter it. When an enter action is performed to a credential the memtype will erase the credential name from the focus and proceed to type the username, then the hop, the password and the submit strings of that credential.

If we prefer to see the username or password of that credential instead, we can perform a back movement on the joystick and it’ll enter the detail mode of that credential, printing “`[DETAIL]`”.

5.- Credential Detail

When entered to a given credential detail indicated by the “`[DETAIL]`” message, the username and password can be typed alternatively by the memtype when pressing up or down.
To return to the credential list, the apply joystick movement can be done.
During the detail state if a back joystick movement is done, the memtype becomes locked again just like when it was first connected into the USB.

6.- Managing the memtype with mttool

The mttool program is a command line tool to manage the memtype device.
It can do all the following actions: encrypt and write the credential to the memtype, read them decrypted to the computer, set PINs, load keyboard dictonary tables to the memtype and get memory size and firmware version from the device.

6.1.- Drivers and Installation

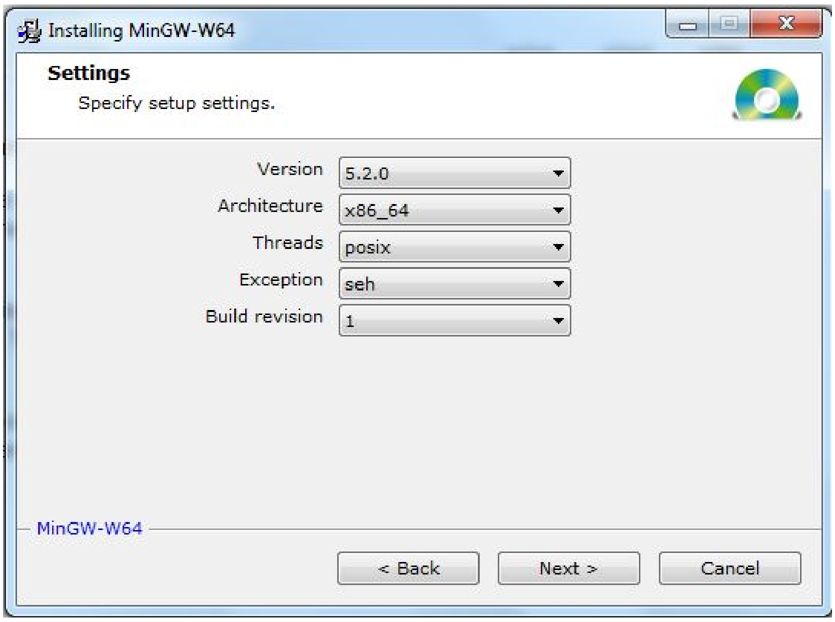
A precompiled binary of the tool can be obtained by downloading the sources from `https://github.com/jim17/memtype/archive/master.zip` and unzipping it, under the memtype-master/mttool/bin directory.

Where statically linked binaries can't be provided make sure to install the library dependencies before executing.

6.1.1.- Windows

Compiling:

1. Install Mingw-w64
2. In your windows machine download: http://sourceforge.net/projects/mingw-w64
3. For 64 bit OS select the following installation options (*if 32 bit OS change Architecture to **i686**.):



Installing MinGW-W64

Settings

Specify setup settings.

Version: 5.2.0

Architecture: x86_64

Threads: posix

Exception: seh

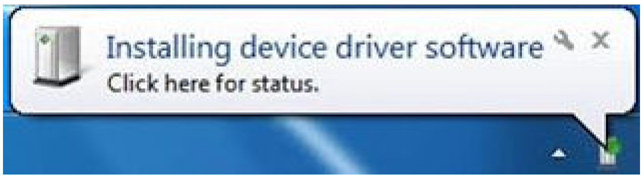
Build revision: 1

4. Add your gcc into windows PATH (example: **C:\Program Files\mingw-w64\x86_64-5.2.0-posix-seh-rt_v4-rev0\mingw64\bin**)

5. Compiling… execute wingen.bat (ensure to remove "_x86" from Makefile if you're compiling on a 64 bit OS).

Driver Installation:

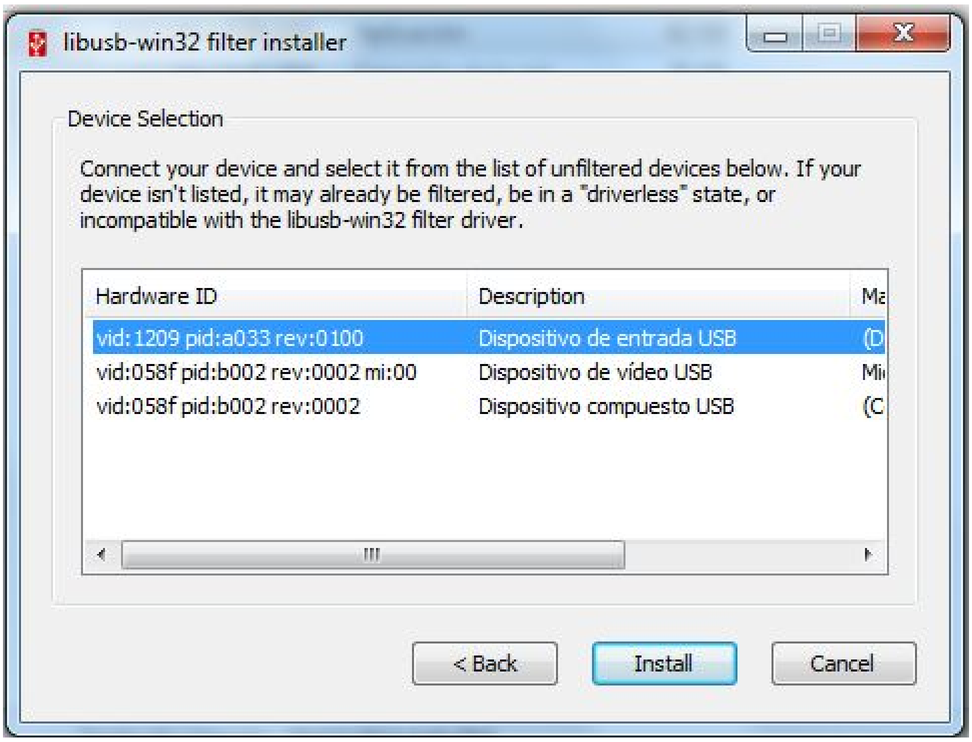
1. Let windows install the default driver for your Memtype.



Installing device driver software

Click here for status.

2. If you want to make use of command line tool with your device you should also do the following steps: (**Download** http://sourceforge.net/projects/libusb-win32/files/libusb-win32-releases/1.2.6.0/libusb-win32-bin-1.2.6.0.zip/download)
3. Move the following files (amd64 for 64 bits or x86 for 32 bits):
   amd64\libusb0.sys to Windows\system32\drivers\libusb0.sys
   amd64\libusb0.dll to Windows\system32\libusb0.dll
4. Execute install-filter-win
5. Select "Hardware ID: **vid:1209 pid:a033**"
6. Click Install



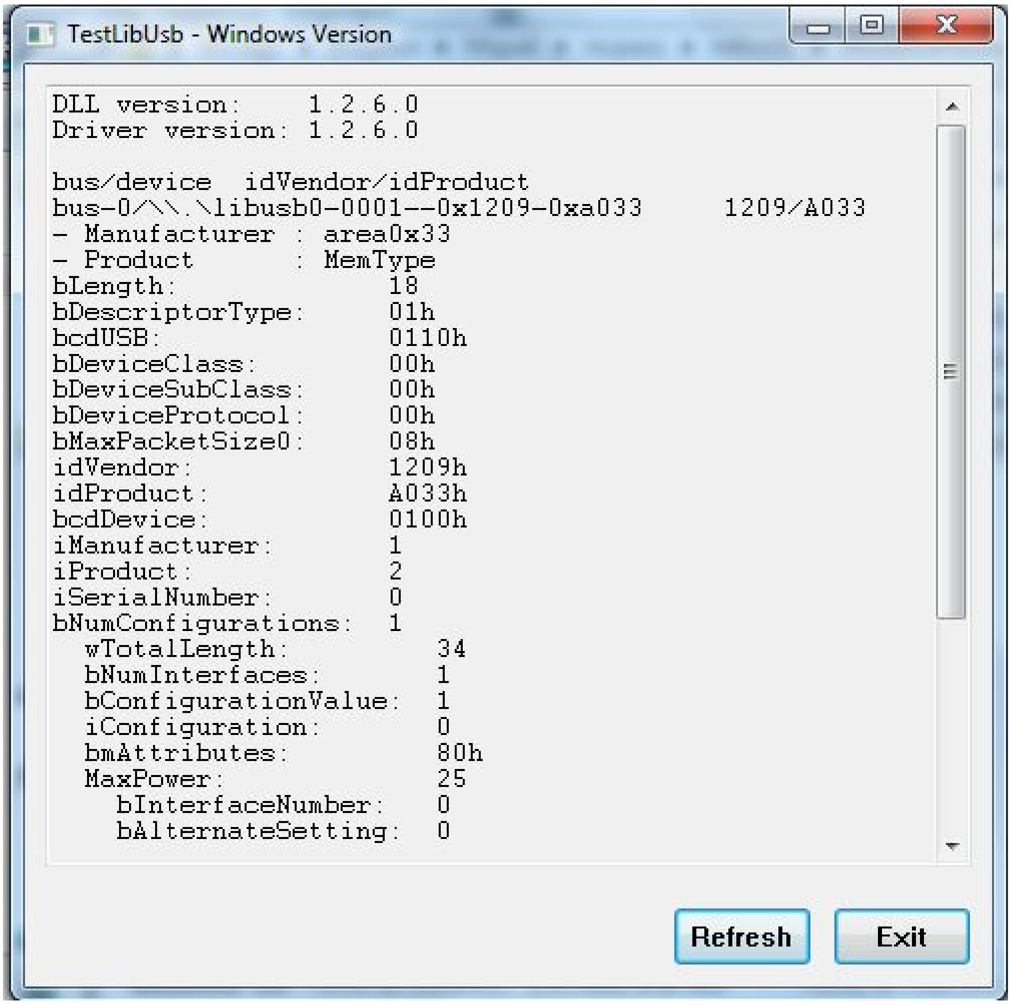
libusb-win32 filter installer

Device Selection

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.

Hardware ID	Description	Ma
vid:1209 pid:a033 rev:0100	Dispositivo de entrada USB	(D
vid:058f pid:b002 rev:0002 mi:00	Dispositivo de vídeo USB	Mi
vid:058f pid:b002 rev:0002	Dispositivo compuesto USB	(C

7. To Verify your driver installation is working execute **testlibusb-win.exe**, you should see something similar to:



```
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:                25
    bInterfaceNumber:      0
    bAlternateSetting:     0
```

8. Congrats !!! Now you should be able to run mttool with your Windows OS.

6.1.2.- Linux

Dependencies:

libusb-dev: `$ sudo apt-get install libusb-dev`

Compiling:

```
$ wget https://github.com/jim17/memtype/archive/master.zip

$ unzip master.zip

$ cd memtype-master/mttool/

$ make
```

Udev rules:

```
$ cd memtype-master
```

```
    $ sudo cp 99-MemType.rules /etc/udev/rules.d/
    $ sudo service udev reload
```

If the memtype was connected on the usb, disconnect and reconnect to apply new udev rules.

Now that the binary is compiled and permissions are right, you can run it:

```
    $ ./mttool
```

6.1.3.- MacOS X

Dependencies:

Using Mac ports:

libusb: `$ sudo port install libusb`

Using Brew:

libusb: `$ sudo brew install libusb-compat`

Compiling:

```
    $ wget https://github.com/jim17/memtype/archive/master.zip

    $ unzip master.zip

    $ cd memtype-master/mttool/

    $ make
```

That'll generate the mttool binary that can be executed by:

```
    $ ./mttool
```

6.2.- Credentials XML file format

Memtype xml credentials file format is a very simple xml file with the following structure:

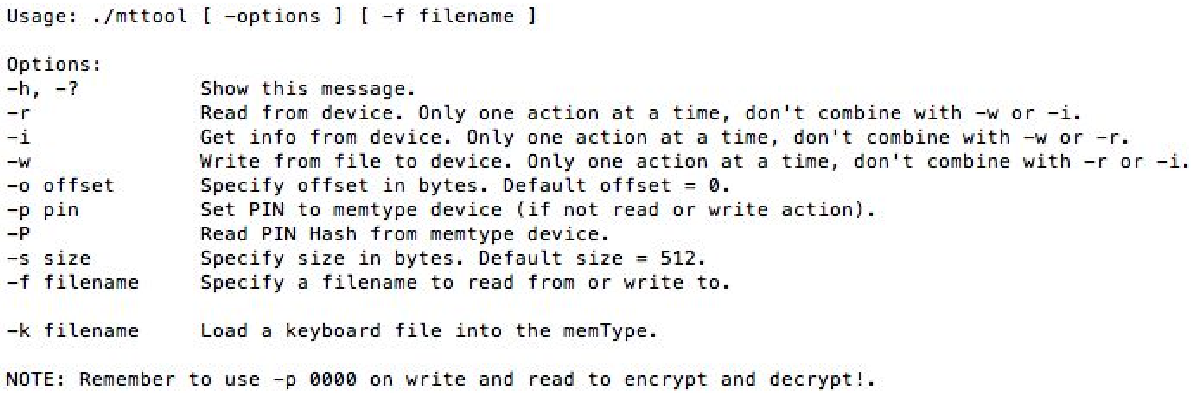
```
<?xml version="1.0" encoding="UTF-8" ?>
<mt>

<cred>
<name>Credential1</name>
<user>user1</user>
<hop> </hop>
```

```
<pass>password1</pass>
<submit>
</submit>
</cred>

<cred>
<name>Credential2</name>
<user>user2</user>
<hop> </hop>
<pass>pass2</pass>
<submit>
</submit>
</cred>

<cred>
<name>Another</name>
<user>AnotherUser</user>
<hop> </hop>
<pass>AnotherPAssword</pass>
<submit>
</submit>
</cred>

</mt>
```

All the tags are needed and they can be empty. The usual hop string is a tabulation character, and the usual submit string is an enter character (in this way you’re automagically logged in when entering the credential inside a website).

6.3.- Examples

When executing the mttool binary a usage help will be printed to the stdout, any user who ever used any other command line program will find it intuitive.



```
Usage: ./mttool [ -options ] [ -f filename ]

Options:
-h, -?          Show this message.
-r              Read from device. Only one action at a time, don't combine with -w or -i.
-i              Get info from device. Only one action at a time, don't combine with -w or -r.
-w              Write from file to device. Only one action at a time, don't combine with -r or -i.
-o offset       Specify offset in bytes. Default offset = 0.
-p pin          Set PIN to memtype device (if not read or write action).
-P              Read PIN Hash from memtype device.
-s size         Specify size in bytes. Default size = 512.
-f filename     Specify a filename to read from or write to.

-k filename     Load a keyboard file into the memType.

NOTE: Remember to use -p 0000 on write and read to encrypt and decrypt!.
```

6.3.1.- Reading the credentials

To read the credentials from the memtype to a xml file:

```
$ ./mttool -r -f tst.xml -p 0000
```

`-r`	Read from the memtype.
`-f tst.xml`	Use specified output file.
`-p 0000`	Use specified pin.

6.3.2.- Writting credentials

To write a credentials list in xml file format to the memtype device:

```
$ ./mttool -w -f tst.xml -p 0000
```

`-w`	Write to the memtype.
`-f tst.xml`	Use specified input file.
`-p 0000`	Use specified pin.

6.3.3.- Setting the PIN

The PIN is used to encrypt and decrypt the credentials in the memtype memory, so before changing it the content of the memtype should be read to the computer, and after setting the new PIN it should be written back to the memtype with the new PIN.

```
$ ./mttool -p 0000
```

6.3.4.- Getting info

To get information about the memtype device:

```
$ ./mttool -i
```

It returns the firmware version and memory size in bytes of the device in xml format, it can also be written to a file with the file flag.

6.3.5.- Changing the keyboard definition

The dictionary table that is used to convert from characters to keycodes has to be the one matching the keyboard layout on your computer (or other device) where you'll use the memtype.
That way the memtype knows the right keycodes to send for writting the strings it has on memory.

```
$ ./mttool -k en_US_mtkbd
```