USB Swiss Armory Pocket Knife

Reinventing the USB Armory to be Cool
(At least to me)

Summary

At Defcon 26 I made some impulse purchases which I almost immediately regretted. One of them was the USB Armory which I thought was going to be much more than it was. I mean, it's cool for a couple years ago, but it just wasn't working for me. So I put time into a design to make it into what I really wanted: an EDC device that does a little bit of everything. What you see before you is my version of an EDC device that does small security tasks. The idea here is to have something easily pocketable that does many tasks when a stronger rig is unavailable to you.

The intended use cases are:

- A Kali image for basic tasks. Just whatever I found useful for app and net testing.
- Encrypted partition for findings and whatever I might need to transport. I opted out of using Interlock, but copy files to and from a "burn basket" folder.
- Dual boot to quickly switch modes. Plugging in and out will toggle between two modes: from Kali to a variety of specialized utility modes:
    - *Responder*. Places the USB armory act as a local USB network device to steal creds.
    - *Passive Network Discovery*. Using Kismet and host mode, scans and logs networks for warwalking, wardriving, etc.
    - *Active Network Discovery and Attacks*. Using besside-ng network keys are actively sought for later cracking in John. Specific IPs may be targeted
    - *Basic John password crack*. Using Byepass and some specialized rules, a generalized (but non-optimized) attack is made on a password hash. The idea is that the hash is placed on the USB armory and then plugged into a convenient adapter to solve offline. Yes, yes, yes I *know* it is not optimized – but it is not meant to be. This is a pocket knife, not a full rig, so it is meant to be easy and quick to deploy.
- Each of these modes are configurable with simple start commands

You will need:

- A compatible Wifi adapter. I use the TP-Link WN722N version 1 (make sure you have the right version or it won't work)
- A host adapter

Device Creation

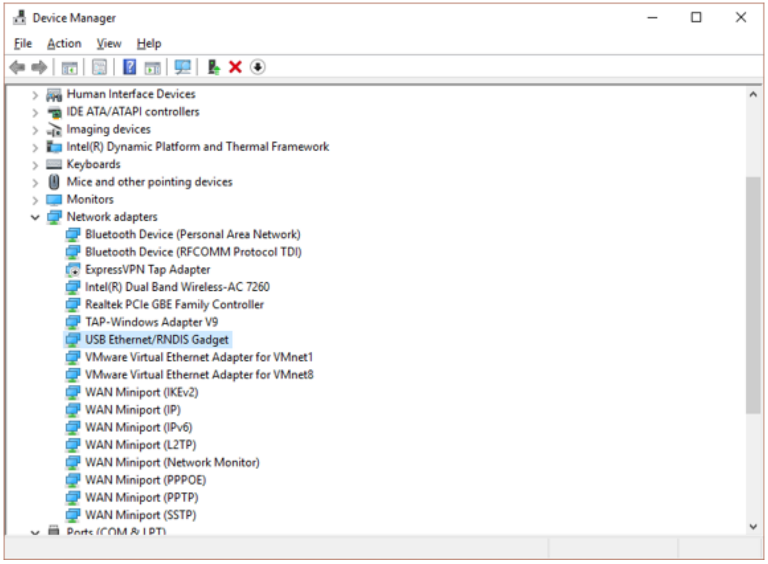
When I first started exploring InfoSec, it drove me crazy to find projects that were super interesting but had missing commands that I had to spend late nights figuring out. Let me know if you see that here so it can be corrected.

Step 1: Basic Kali Image

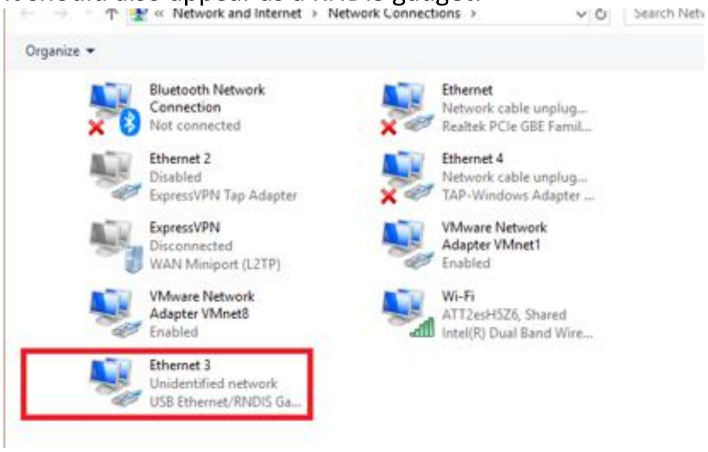
1. Start out with a good SD drive and get initial image.
    a. Follow the instructions posted on USB Armory Google Groups except:
    b. The poster states to use Ext2Fsd in Windows. My experience shows that this need to be made from a Linux PC. The program will not save any changes made on Windows (at least not on 10).
    c. Run gparted to partition the drive. Keep the mmcblk0p2 partition since that will be our encrypted drive.
    d. Do not use the driver that the poster recommends.
2. If you have Windows 10, you will have to make some changes so the USB Armory will be seen:
    a. Change owner of C:\Windows\INF\rndiscmp.inf:
        - On the file itself: Properties > Security > Advanced
        - Change owner
        - Close dialog, reopen, assign Full Control
    b. Make the following changes:
       C:\Windows\INF\rndiscmp.inf

```
…
[RndisDevices.NTamd64]
;when MSID is used for RNDIS Ethernet devices over USB transport
%RndisDevice%    = RNDIS.NT.6.0, USB\MS_COMP_RNDIS&MS_SUBCOMP_5162001
;when CompatID is used for RNDIS Ethernet devices over USB transport
%RndisDevice%    = RNDIS.NT.6.0, USB\Class_EF&SubClass_04&Prot_01
;for USB armory
%RndisDevice%    = RNDIS.NT.6.0, USB\Class_02&SubClass_02&Prot_FF
…
```

3. Verify USB Armory is seen in Device Manager under Network Adapters:


Device Manager
File
Action
View
Help
Human Interface Devices
IDE ATA/ATAPI controllers
Imaging devices
Intel(R) Dynamic Platform and Thermal Framework
Keyboards
Mice and other pointing devices
Monitors
Network adapters
Bluetooth Device (Personal Area Network)
Bluetooth Device (RFCOMM Protocol TDI)
ExpressVPN Tap Adapter
Intel(R) Dual Band Wireless-AC 7260
Realtek PCIe GBE Family Controller
TAP-Windows Adapter V9
USB Ethernet/RNDIS Gadget
VMware Virtual Ethernet Adapter for VMnet1
VMware Virtual Ethernet Adapter for VMnet8
WAN Miniport (IKEv2)
WAN Miniport (IP)
WAN Miniport (IPv6)
WAN Miniport (L2TP)
WAN Miniport (Network Monitor)
WAN Miniport (PPPOE)
WAN Miniport (PPTP)
WAN Miniport (SSTP)
Ports (COM & LPT)

It should also appear as a RNDIS gadget:



« Network and Internet › Network Connections ›

Search Netw

Organize ▾

Bluetooth Network Connection
Not connected

Ethernet
Network cable unplug...
Realtek PCIe GBE Famil...

Ethernet 2
Disabled
ExpressVPN Tap Adapter

Ethernet 4
Network cable unplug...
TAP-Windows Adapter ...

ExpressVPN
Disconnected
WAN Miniport (L2TP)

VMware Network Adapter VMnet1
Enabled

VMware Network Adapter VMnet8
Enabled

Wi-Fi
ATT2esH5Z6, Shared
Intel(R) Dual Band Wire...

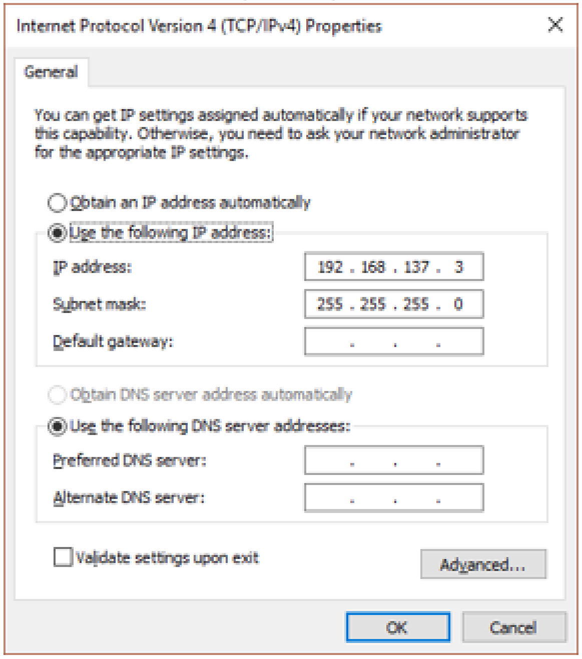
Ethernet 3
Unidentified network
USB Ethernet/RNDIS Ga...

If it doesn't:

a. Verify steps in https://github.com/inversepath/usbarmory/wiki/Host-communication#setup--connection-sharing-windows-7-8-10 - including top part
b. Try a reboot.

When working fully you'll not only see it in above screens, but also on tray as Ethernet connection.

4. Make the following settings in IPv4 for the USB armory:



Internet Protocol Version 4 (TCP/IPv4) Properties

General

You can get IP settings assigned automatically if your network supports this capability. Otherwise, you need to ask your network administrator for the appropriate IP settings.

○ Obtain an IP address automatically
● Use the following IP address:

IP address: 192 . 168 . 137 . 3
Subnet mask: 255 . 255 . 255 . 0
Default gateway: . . .

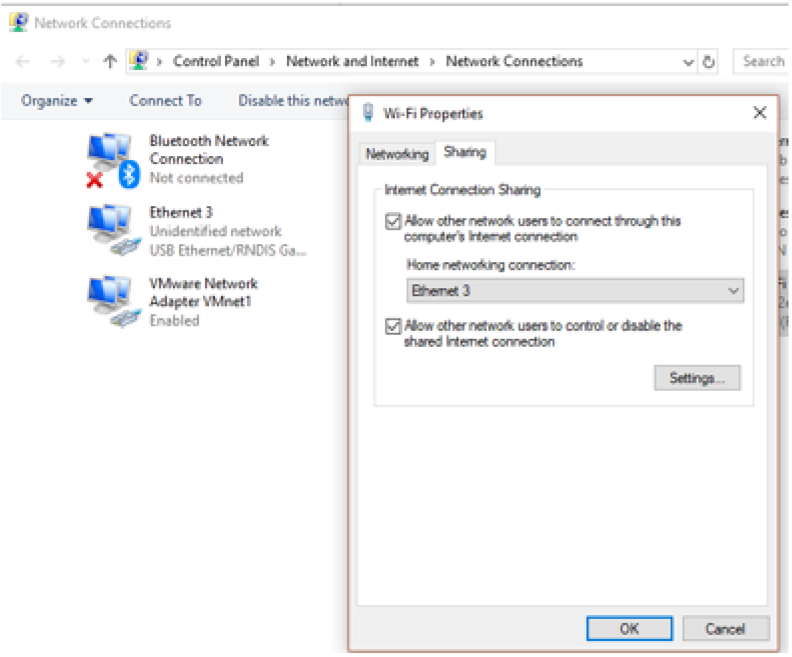
○ Obtain DNS server address automatically
● Use the following DNS server addresses:

Preferred DNS server: . . .
Alternate DNS server: . . .

□ Validate settings upon exit

Note: Do not set a preferred DNS.

5. Set for Internet sharing:



☑ Allow other network users to connect through this computer's Internet connection

Home networking connection: Ethernet 3

☑ Allow other network users to control or disable the shared Internet connection

Note: you may have to uncheck and recheck this when you disconnect the USB armory

Checkpoint: Verify you can:

- Ping the USB armory at 192.168.137.2
- Use putty to connect to the USB armory
- Ping from within the USB armory to hosts on the network and Internet addresses.

Step 2: Basic Kali Image

1. Change / verify the sources in /etc/apt – it is very likely blank. A basic search of what it should like will give you the correct address.
2. Verify connection by **apt-cache search**, which will show all programs
3. Run **apt-get update**
4. Run the following to get custom image of tools (you may choose others): **apt-get install -y hash-identifier wfuzz metasploit-framework sslscan nikto dirb abootimg cgpt fake-hwclock ntpdate u-boot-tools vboot-utils vboot-kernel-utils aircrack-ng cewl crunch dnsrecon dnsutils ethtool exploitdb hydra john libnfc-bin medusa metasploit-framework mfoc ncrack nmap passing-the-hash proxychains recon-ng sqlmap tcpdump theharvester tor tshark usbutils whois windows-binaries winexe wpscan apache2 atftpd haveged openssh-server openvpn tightvncserver cryptsetup isc-dhcp-server lvm2 wpasupplicant cowpatty reaver wifi-honey ettercap-text-only bulk-extractor volatility hydra sqlmap backdoor-factory wipe**

Checkpoint: Verify you can:

- Run something fun like metasploit or nmap
- Take break – you now have a gadget that is useful, just need to improve it.

Bonus Points: Give your Armory a good code name and change /etc/motd to show it in ASCII art. I use the generator at patorjk.com/software/taag/.

Step 3: Add VID and PID

To ensure that the device is recognized on all systems as a real RNDIS gadget, you need to specify a VID and a PID, otherwise Windows 10 (and possibly other systems) will infuriatingly insist that it is a USB Serial Device.

1. Backup /etc/modprobe.d/usbarmory.conf
2. Change the configuration to add the items in bold:

   <u>usbarmory.conf</u>
   ```
   options g_ether use_eem=0 dev_addr=1a:55:89:a2:69:41 host_addr=1a:55:89:a2:69:42
   idVendor=0x04b3 idProduct=0x4010
   # To use either of the following, you should create the file /disk.img via dd
   # "dd if=/dev/zero of=/disk.img bs=1M count=2048" would create a 2GB disk.img file.
   #options g_mass_storage file=disk.img
   #options g_multi use_eem=0 dev_addr=1a:55:89:a2:69:41 host_addr=1a:55:89:a2:69:42
   file=disk.img
   ```
   Note: the options line is all one line including the VID and PID.
3. Reboot device

Checkpoint: Verify you can still SSH into the device.

Bonus Points: Specify your own PID and VID – always a good idea if you are attempting to evade forensics.

Step 4: Create Encrypted Partition

Your most obvious option for encryption on USB Amory is Interlock. And it can be frustrating to setup off menu. One key thing to know is that you cannot have the sahara module running while using LUKS. So what we’re going to do is set up the partition in LUKS so that it can be used later with Interlock if you want. Our standard usage will be to access LUKS as needed and place any working documents in a burnbasket folder. A shutdown procedure will wipe that basket clean as needed.

1. To determine partitions use lsblk. Our target is mmcblk0p2:
   ```
   root@xxxxxxxx:~# lsblk
   NAME        MAJ:MIN RM  SIZE RO TYPE MOUNTPOINT
   mmcblk0     179:0    0   30G  0 disk
   |-mmcblk0p1 179:1    0 20.6G  0 part /
   `-mmcblk0p2 179:2    0  9.3G  0 part
   ```
2. Make an encrypted volume – I made one for 9gb below:
   ```
   pvcreate /dev/mmcblk0p2
   vgcreate lvmvolume /dev/mmcblk0p2
   lvcreate -L 9G -n encryptedfs lvmvolume
   ```
3. Turn off sahara and verify it is not on. Skip this step at risk of your frustration – it simply won’t work with sahara loaded.
   ```
   root@xxxxxxxx:/lib/modules# rmmod sahara
   root@xxxxxxxx:/lib/modules# lsmod | grep sahara
   ```
4. Now time to setup the LUKS volume.
   ```
   root@xxxxxxxx:/lib/modules# cryptsetup luksOpen /dev/mmcblk0p2 interlockfs
   Enter passphrase for /dev/mmcblk0p2:
   root@xxxxxxxx:/lib/modules# cryptsetup status interlockfs
   /dev/mapper/interlockfs is active.
     type:    LUKS1
     cipher:  aes-xts-plain64
     keysize: 256 bits
     key location: dm-crypt
     device:  /dev/mmcblk0p2
     sector size:  512
     offset:  4096 sectors
     size:    18868224 sectors
     mode:    read/write
     root@xxxxxxxx:/lib/modules# dd if=/dev/zero of=/dev/mapper/interlockfs
   ```

5. Write down this password in multiple places: encrypted volume on your laptop / phone, and a backup copy as well. Seriously – do this now.
6. Create /home/burnbasket. This will be for all temporary files to be worked upon and then deleted on usbdown
7. In /local/usr/bin, add the following three files (these are already written for you in the *Scripts and Configurations* directory of this GitHub repository):

   unlockfs

   ```
   #!/bin/bash
   # Unlock out 9g encrypted volume
   echo Opening encrypted file system…
   rmmod sahara > /dev/null 2>&1
   cryptsetup luksOpen /dev/mmcblk0p2 encryptedfs
   mkdir /encryptedfs/
   mount /dev/mapper/encryptedfs /encryptedfs
   cd /encryptedfs
   echo Ready.
   echo
   ```

   lockfs

   ```
   #!/bin/bash
   # Safely shutdown the encrypted volume
   echo Looking encrypted file system…
   cd /
   umount /encryptedfs > /dev/null
   rm /encryptedfs 2>&1 > /dev/null 2>&1
   cryptsetup luksClose encryptedfs
   echo Complete
   echo
   ```

   usbdown

   ```
   #!/bin/bash
   echo Wiping files...
   wipe -f -s /home/burnbasket/*.* > /dev/null 2>&1
   sync
   echo Shutting down...
   ledswitch 3
   sleep 1
   shutdown -P now
   ```

   Remember to chmod +x on each of these files

Checkpoint: Verify you can:

- Run unlockfs and access /encryptedfs
- Run lockfs and verify that /encryptedfs mount is dismounted.
- Run usbdown and make sure the device wipes files from the burnbasket folder and turns off the LED

If you can do all of these, you now have a portable security tester with a secure volume – not bad. Might want to take a break before hitting the next part.

Step 5: Dual Boot

Our next step will make the gadget a true multifunctional device. We will alter between the Kali and encrypted file store and an attack mode. To know which mode we are in, we will use the LED:

- For the Kali / encrypted storage: a heartbeat
- For the attack mode: a flash every second

The entire idea was based on the google group idea:
https://groups.google.com/forum/#!msg/usbarmory/lfpfKsRZvQI/OpD6yqhZAwAJ

What we have done so far is access the USB armory in USB mode. The steps below change it so that it goes into a slightly different USB mode to run responder as well as run in host mode. These are determined by device tree blobs (DTBs); fortunately the Kali image has the hard work complete already and has the DTB file ready for us. We just need to switch between them.

Make sure to do all of these steps completely, and it might be a good idea to backup what you've done so far. If you run into trouble, eject the SD from the device and modify it so you can get back to a previous step.

1. Make a copy of the original
   ```
   cp /boot/imx53-usbarmory.dtb /boot/imx53-usbarmory-orig
   ```

2. Make sure you can run the following commands – if not you are missing one or more installs:
   - kismet
   - besside-ng
   - john
   - responder

3. Copy each file from *Scripts and Configurations* in this GitHub repository.
   - The configuration files are those that will setup the tools the way that will work with the USB armory and / or the scripts
     Note: The john configuration uses the best126 rule from https://github.com/webpwnized/byepass. You may want to alter this rule based on your needs or get additional dictionaries – this is a highly recommended repository.
   - Remember to chmod +x each script file (not all have a .sh extension!)

4. Download the rockyou password dictionary from https://wiki.skullsecurity.org/Passwords to /usr/share/wordlists

5. Understand where everything is placed:
   - The dual boot takes place in /etc/rc.local
   - The payloads script is /payloads/main.sh – something you might want to tinker with
   - Basic commands are placed in /usr/local/bin

Checkpoint: Verify that the device:

- Performs a correct dual boot. By plugging it in once, it should go to heartbeat mode. By removing it and pluggin back in, it should go to a one-second pulse mode.
- Commands are running. Try **usbcmds** to give it a quick test.

If you are successful, then you just may have a working portable device fully ready to go!

Bonus Points: Copy some handy Windows tools to your home directory so you can have those with you. I carry mimikatz, Windows nmap, pstools, hashcat, and my other github tools (https://github.com/saintcrossbow).

Device Usage

Toggled Boot

On each boot the device will toggle between Kali and the attack mode. A heartbeat indicates Kali mode, a one-second pulse indicates in attack mode.

Basic Usage

Kali

- Access the device at 192.168.137.2 via Putty.
- To show a cheatsheet of the commands: **usbcmds**
- Accessing the encrypted file system:
    - **unlockfs** to access drive
    - **lockfs** to shut down drive
    - **cryptsetup luksAddKey /dev/sda3** to add a new key to your encrypted drive

  There are plenty of other cryptsetup commands of which you'll want to take advantage. Read up on your options before relying entirely on the drive.
- Remember to always use **usbdown** to shutdown the device gently (which also wipes the burnbasket folder).
- Change MAC address in /etc/modprobe.d/usbarmory.conf
- To make sure the next boot starts in Kali mode again, use **startusb** (and if you change your mind, **startresp** will place in attack mode)
- Change the LED pulse or turn it off using **ledswitch**: no argument provides a cheatsheet, 1=heartbeat, 2=timer, 3=silence)
- Set the next attack mode by **altboot** (1=responder, 2=payloads)
- Reset all besside logs by using **resetlogs**
- Set payload for next attack:
    - **paylad-kismet**: passive wifi monitoring
    - **payld-besside**: active wifi attack
        - If no argument is specified, all wifi networks in range are attacked. Make sure you have permission to do this.
        - Specify a BSSID (the MAC) to attack a specific wifi network; e.g. **payld-besside xx:xx:xx:xx:xx:xx**
    - **payld-john**: offline passwork crack

Offline Password Cracking

- Logs and solved passwords in /root/.john
- Place hash / hashes to solve in /home/johnhost/target.txt
- Set to host with payloads by **altboot 2**
- Specify password cracking and format by **payld-crack *format***
- Shutdown using **usbdown** and then plug into any spare USB outlet.

You really need to understand john fully to get the most out of this mode. However this is a good fire and forget option that balances efficiency with likelihood of success. Once the armory stops flashing, it has solved as much as it can.

Responder Mode

1. Set to Responder by **altboot 1**
2. Plug into target machine (with permission of course)
3. Logs are in /etc/responder/logs

Kismet Passive Scan

1. Set to host with payloads by **altboot 2**
2. Set to kismet by **payld-kismet**
3. Shutdown device using **usbdown**
4. Put power to the host adapter, plug in armory on right and wifi adapter on left.
5. Power on and scan for at least 5 minutes to get network details
6. Logs stored in /kismet

Besside Active Attack

1. Set to host with payloads by **altboot 2**
2. Set to besside by **payld-besside**
3. Shutdown device using **usbdown**
4. Put power to the host adapter, plug in armory on right and wifi adapter on left.
5. Attack starts in 5 seconds upon power on.
6. Logs stored in /wpa.cap and /wep.cap

Note that besside attack is very aggressive and noisy. You also – of course – need permission before attacking a network

Recommended Kit Setup

If you make this your everyday carry device, there are a few other items I recommend you carry with you:

1. The original host adapter. You won’t be able to do the WiFi attacks without it.
2. A USB cable to power the adapter.
3. A micro SD adapter to act as first aid for a malfunctioning device. It can happen and usually does at worst time possible.
4. A USB drive with putty and scp so you can access on any PC.
5. A USB hub. Trust me, you’ll wish you had it with you if you don’t pack it.

Below is a picture of my EDC setup (which also has the addition of a WHID Cactus):