Securing your Laptop like you mean it: Virtualization Based Security

Milosch Meriac,
Principal Hardware Security Researcher
@ xen1thLabs, DarkMatter Abu Dhabi
My Open Software & Hardware Projects

Shared fun is twice the fun

I created a range of open hardware designs and software tools around RF(ID)/BLE security research and electronic art projects. You can find a more information on my work at meriac.com
OpenBeacon.org
Realtime 2.4GHz
Localization & Human Interaction Analysis, see also SocioPatterns.org

Blinkenlights Stereoscope
960 x Realtime 2.4GHz
Wireless Halogen Dimmers for Toronto City Hall

Xbox Linux Core Team
Breaking the first trusted computing platform for consumers
Arm Mbed uVisor Security

mbed OS
- mbed OS is a modular, secure, efficient, open source OS for IoT
- Connects to mbed Device Connector

<table>
<thead>
<tr>
<th>mbed OS Core</th>
<th>mbed OS Drivers</th>
<th>mbed OS uVisor</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application Code</td>
<td>Communication Management</td>
<td>Secure Drivers</td>
</tr>
<tr>
<td>Libraries</td>
<td>mbed Client</td>
<td>uVisor</td>
</tr>
<tr>
<td>mbed OS API</td>
<td>mbed TLS</td>
<td>Hardware Interfaces</td>
</tr>
<tr>
<td>IP Stack</td>
<td>BLE API</td>
<td>ARM Cortex-M CPU</td>
</tr>
<tr>
<td>Thread API</td>
<td>BLE</td>
<td>MPU</td>
</tr>
</tbody>
</table>

uVisor secure isolation

Arm Ltd
Led mbed OS uVisor code compartmentalization project for microcontrollers. Uses hardware isolation features of Arm v7M/v8M for isolated execution.
Introduction:
Laptop Threat Model

Follow The White Rabbit
Malicious firmware weakens the operating system security at boot …
... or runs in the background creating active backdoors – undetectable by the OS and persistent across OS reinstalls
Attackers physically obtaining temporary access to hardware can result in irrecoverable loss of control over your hardware.

Watch a Hacker Install a Firmware Backdoor on a Laptop in Less Than 5 Minutes

This demo shows that “evil maid attacks,” hacks where an attacker has physical access to a target computer, are not as complicated as you may think.

By Lorenzo Franceschi-Bicchieri
Jul 23, 2016, 11:27pm

Hacker lore is littered with tales of mysterious attackers breaking into hotels—perhaps at a conference—to get their hands on someone’s
Supply chain attacks enable pre-installation of backdoors for targeted users – most attacks won’t require installation of physical chips like in this example.
Simple malware runs on the main CPU (various system- & PCI BIOS NVM chips) - hidden in SMM/HV.

Advanced malware permanently installs on embedded controllers like Keyboard, Hard disk, SSD, Network, Intel AMT/ISM, WIFI, BT, WAN, Thunderbolt-adapters, SD-Card or USB devices like webcams and disks.
Modern OSs like Windows and selected Linux distribution detect installed hardware, update device & system firmware, update vendor-specific software and install binary drivers.
Should we just capitulate?
Can I trust my Hardware?

Speedrun: How bad is hardware complexity?
Hardware Platform Example:

**Lenovo ThinkPad Carbon X1**

2019 Lenovo X1 Carbon (7th Gen)

- **Screen**: 14.0 inch, FHD matte (400 nits), FHD touch (400 nits), FHD ePrivacy (400 nits), WQHD matte (300 nits), UHD 10-bit with HDR (500 nits)
- **Processor**: up to Intel Whiskey Lake i7-8565U (4 cores, 8 threads @ 1.8-4.6GHz)
- **Video**: UHD Graphics 620 (integrated)
- **Memory**: up to 16GB LPDDR3 (soldered, dual-channel)
- **Storage**: 1x M.2 80 mm NVMe OPAL2, up to 2 TB
- **Connectivity**: Intel WiFi 9650 with Bluetooth 5.0 (?), Intel Ethernet Connection, optional WWAN
- **Ports**: 2x USB-C Thunderbolt 3, 2x USB-A 3.1, HDMI 1.4(?), SIM/MicroSD, doc-port, headphone/mic, Lock
- **Battery**: 51 Wh, 65W power adapter (USB Type-C)
- **Size**: 323 ot 12.71″(W) x 217 x 8.54″(D) x 14.95 or 0.58” mm (H)
- **Weight**: from 1.11 kg / 2.46 lbs (+ power supply)
- **Extras**: IR cameras with ThinkShutter, finger-sensor, quad speakers
Hardware Platform Example: Lenovo ThinkPad Carbon X1

- Soldered DDRAM memory complicates cold-boot attacks.
- DDRAM has footprint for adding a metal shield
- Detection of M.2 SSD removal
- Free slot for custom security module (WAN-slot) and SIM-Card slot for secure SIM card
- Hard-Off-Switch, reachable via pinhole in case
Hardware Platform Example: Lenovo ThinkPad Carbon X1

- Discrete TPM module
- TPM buffered by dedicated CR2032 coin cell
- Anti-Tamper switch protecting the TPM
- Battery-removal detection
Hardware Platform Example: Lenovo ThinkPad Carbon X1

- Two serial NVM flash memories
Hardware Platform Example: Lenovo ThinkPad Carbon X1

- SIM-Card slot for secure SIM card - connected to WAN modem
- Various microcontrollers, including (Microchip, STM)
Hardware Platform Example: Lenovo ThinkPad Carbon X1

- Metal housing for added electromagnetic fault injection & side channel attack resilience
- Fingerprint Sensor
Getting Trust Back*

*Decent security with reasonable compromises*
“Anyone can create a security system that they themselves can’t break”

Schneier’s Law
... so let’s try designing a secure laptop anyways!

What Do We Want?

- Unsuspicious look & UX
- Ultra-portable
- Secure & Trusted connectivity (USB & Net)
- Create user trust & security through virtualization
- Flexible sourcing & trusted repair
- Minimized attack surface
- Physical Crypto-Token to lock laptop:
  - User friendly security & Multifactor Authentication

“Anyone can create a security system that they themselves can’t break”

Schneier’s Law
How to get decent security with reasonable compromises

1. Supply-Chain
   - Buy hardware in brick & mortar store to avoid targeted attacks
   - Depending on threat model – buy in a different city or country.
   - Pay cash to hide knowledge about obtained hardware
   - Physical Hardening

2. Secure Boot
   - If available for your platform – use CoreBoot BIOS replacement
   - For most modern platforms though UEFI with secure boot enabled and hardened BIOS settings is fine.
   - Settings, System and Data protected by boot process

3. Hardened OS
   - Secure booting of minimal headless Linux OS
   - Tiny: Minimal executables, modules and boot process in immutable & encrypted filesystem
   - Launches IOMMU-protected hypervisor

4. Untrusted OS
   - Untrusted virtualized OS receives direct access to the GPU (IOMMU protected)
   - All other devices are handled by the hardened OS, virtualized for the untrusted OS
   - Disk-encryption, network & power handled by Linux OS
Laptop Security: Implementation Comments

- Protect user by making laptop look normal – don’t encourage theft by making it look “special”.
- Ensure low weight to make Laptop as portable as possible to ensure that user doesn’t leave laptop unattended.
- Use strong multifactor authentication for unlocking laptop: protecting against password leaks/guessing.
- Use hardened Linux to protect the Windows instance running in a hypervisor:
  - Allows trusted enforcement of policies (USB camera, network, WIFI, Bluetooth) outside of Windows by Linux.
  - Filter/authenticate/protect network traffic in Linux before passing it to Windows instance.
  - For high security systems we suggest to force all network traffic through a VPN – allowing centralized services, network security and malware detection.
- Remove dependency on single supplier (software and hardware) to enable scalable security:
  - security model and policies must be transferrable to later devices
  - suppliers and must not create a single point of failure
- Configure security-related settings like anti-tamper, secure boot, boot order, DMA security and Execution Prevention – and set Supervisor and User passwords.
- Remove dependency on untrusted 3rd party repair centers:
  - Benefit from availability of spare parts by choosing a well-known vendor of business laptops.
  - This requirement allows the user’s IT department to replace common parts without relying on external services.
  - Results in simplified supply chain security.
- Establish trust by patching/configuring internal BIOS to minimum attack surface – utilizing TPM features and secure boot process to bring up encrypted & hardened Linux system.
- Store root of trust and crypto keys to external USB Hardware Security Module (HSM) that is removed by user whenever laptop is unattended – internal disk can’t be decrypted by an attacker without controlling external USB key and PIN for unlocking the stored keys.
- Enable integrated LTE communication to avoid WIFI access points whenever needed (VPN blocking on WIFI etc).
- Disable external interfaces with DMA bus-master access or DisplayPort as IOMMU security might be bypassed through source spoofing on Thunderbolt or similar interfaces: Prevents use of docking stations – but increases security substantially.
Use nail polish with glitter or other microstructures to discourage & detect attacks: Use “blink test” image comparison

Tamper Evident Glitter Nail Polish

One way to detect physical intrusion attempts on your mobile device is to use nail polish like Fuzzy Coat as a tamper-evident marker on the screws. This was suggested by Eric Michaud and Ryan Lackey at 30C3 and my laptops have had this cheap protection applied prior to going to 30C3. Be sure to take good closeup photos once it has dried so that you can do “blink tests” to verify if the random pattern has changed.

Normal glitter polish works as well, although the features are smaller and have lower contrast than the Fuzzy Coat shown above. You can do your nails with it, too, and look fabulous while you’re coding.

Categories: Security  2014
Reducing attack surface by using hypervisor virtualization for system integrity is a well-established security mechanism for increasing software security and system trust: see Cubes OS, Windows Defender System Guard and MirageOS.
Virtualization Based Security with Linux for greater control

- Run Windows in Hypervisor Compartment, protecting Laptop from Windows-Malware becoming persistent in hardware
  - Windows 10 runs in a hardware-secured hypervisor domain (Intel VT-d, IOMMU)
- External USB Crypto Token used as a 2nd factor for decrypting disk partitions
  - when unplugged, Laptop locks down.
- External Interfaces like LTE, WIFI, USB and Ethernet are protected and controlled by hardened Linux
  - Network traffic tightly controlled by Linux, option to enforce VPN tunnel for all traffic.
  - Policies for communication interfaces controlled outside Windows in Linux
- Internal Windows disks and Linux partitions are encrypted and authenticated by hardened Linux
  - dm_crypt/dm_verity for Linux system partition
  - dm_crypt for data partitions
  - dm_crypt for hypervisor volumes containing Windows disk images
- User password used to decrypt partitions as the 1st factor
- Internal dTPM Security Modules used as 3rd factor to decrypt system partitions, tying encryption into built-in anti-tamper feature of the laptop
- Passive and active tamper countermeasures added to Laptop where required.
How to get decent security with **reasonable** compromises

“Invisible” Linux controls:

- IOMMU configuration against DMA busmaster attacks
- USB device access
  - Webcam
  - Fingerprint
  - WAN Modem
  - Touchpad
- Keyboard & Trackpoint
- Microphones & Speakers
- Network Traffic
  - VPN
  - Firewall
  - WIFI
- LUKS Disk Encryption and TCG OPAL v2.0
- External Crypto Modules
- All hardware, but the GPU
- Integrity of System Partition
- Power States
- Secure Token & Boot
- dTPM Measurements
How to get decent security with reasonable compromises

Virtualized Windows installation controls:
- Direct access to graphics card and GPU, but no access to card BIOS – which is only emulated statically from a file.
- All other PCI peripherals are either simulated as part of a virtualized PC mainboard ...
- ... or use signed VirtIO windows drivers for accelerated access
  - Network
  - Disk IO
- Virtualization enables cool features like snapshots of the VM
  - Reverting to previous named VM snapshots possible
- Enables "Plausible deniability System"
Virtualization Security: Implementation Details - I

1. Windows 10 Pro Installation protected by Linux Hypervisor Hardware Virtualization (QEMU, XEN)
2. No access to real hardware except graphics card - virtualized hardware for all other peripherals: resulting in accelerated 3D-graphics in Windows with native vendor graphics drivers in Windows (nVidia, Intel Graphics)
3. Busmaster-DMA access of graphics card to hardened Linux is prevented by IOMMU
4. Dynamic control of access to critical peripherals like USB camera and built-in microphones handled by Linux-based policies - dynamically hiding them from Windows when inactive.
5. Full control over all external communication (network, user interaction and USB) of the Windows installation by hardened Linux system
6. Protect all hardware peripherals and system BIOS against windows-based malware becoming resident by compromising their firmware through malicious updates
7. Linux power state control and disk encryption allow full control of system security at rest – complemented by external secure USB token
8. Virtualization enables national crypto algorithms of the Windows disk in underlying hardened Linux - protected against leakage by a potentially compromised Windows...
1. Laptop boots into 128MB read-only SSD partition (TCG OPAL v2.0)
   a) Secured by UEFI secure boot & PKI signatures
   b) Early-boot DMA attacks prevented by BIOS settings
   c) Linux Kernel and Initial Ramdisk (InitRD) boot protected by PKI based public key signatures

2. Double-encrypted disk to prevent storage hot-swap attacks:
   a) TCG Opal (v2.0) AES-XTS 256 bit SSD hardware encryption as baseline for baseline tamper-protection
   b) LUKS 2.0 Disk Encryption with Argon2i key derivation for security

3. Password-entry dialog allows entry of two passwords, which are derived into two keys:
   a) kD: Access to plausible-deniability windows system without confidential documents (password used on border controls and hardware-seizure of laptop).
      Upon reboot, this system discards all user changes and data.
   b) kR: Access to work system with confidential documents, only used when user is in a safe environment.
      This system keeps user changes/data across reboots.
   c) Knowledge of the decoy-password doesn’t lead to disclosure of work system data.

4. Memory-hard Argon2i key-derivation used to combine dTPM2.0 secret and user-passwords used to decrypt disk secret token slots (kD or kR) to unlock Opal Disk Encryption

5. Additional key-derivation steps used for generating LUKS2 password for decrypting Linux partitions – involving the secure USB token (required for kR, optional for kD).

6. Forensic analysis of system will not reveal that two Windows System alternatives exist or that the USB key is required for one of them

7. Without access to the secure USB token, no access to the work system is possible while the system is suspended

8. For Windows, the Linux disk encryption is invisible: Windows malware can’t access the disk encryption keys even by compromising the Windows Kernel.
Plugging things together
Implementation: Here’s the working virtualization configuration Makefile for running Windows 10 virtualized, but with full 3D acceleration on the Lenovo Carbon X1 7th gen:

```makefile
# Input files
ISO_WINDOWS10=images/iso/Win10_1903_V1_English_x64.iso
ISO_VIRTIO=images/iso/virtio-win-0.1.171.iso
VIDEO_ROM=images/rom/intel-uhd620.rom
BIOS_ROM=/usr/share/seabios/bios.bin

# Output files
DISK_IMG_SIZE=128G
DISK_IMG_FILE=images/disk/windows.qcow2

.PHONY: prepare run clean
all: $(DISK_IMG_FILE)

run:
  sudo qemu-system-x86_64 \  
    -enable-kvm -M pc -m 8G -cpu host,off,hv_vendor.id=null,-hypervisor \  
    -device vpio-pci,host=00:02.0,bus=pci.0,addr=0.2.0,multifunction=on,x-vga=on,x-igd-gms=4,x-igd-opregion=on,rombar=1,romfile=${VIDEO_ROM} \  
    -chardev stdio,id=seabios -device isa-debugcon,iobase=0x4f2,chardev=seabios \  
    -bios $(BIOS_ROM) \  
    -drive file=${DISK_IMG_FILE},cache=none,if=virtio,format=qcow2 \  
    -object input-linux,id=mouse,evdev=/dev/input/by-path/platform-18042-serio-1-event-mouse \  
    -object input-linux,id=keyb,evdev=/dev/input/by-path/platform-18042-serio-0-event-kbd \  
    -device virtio-net-pci,netdev=eth0 -netdev user,netdev=${DISK_IMG_SIZE}user,netdev=eth0 \  
    -device virtio-rng-pci \  
    -rtc base=localtime \  
    -device nec-usb-xhci \  
      -device usb-host,vendorid=0x04f2,productid=0xb6f2 \  
    -vga none -display none \  
    -snapshot

${DISK_IMG_FILE}:
  qemu-img create -f qcow2 $@ ${DISK_IMG_SIZE}
```
Preparation:
Here’s the virtualization configuration Makefile for running Windows 10 installer – using non-accelerated VGA for the installation & connecting the Windows 10 Setup Disk & the virtual IO drivers for virtual network and virtual disks

```bash
${DISK_IMG_FILE}:
    qemu-img create -f qcow2 $@ ${DISK_IMG_SIZE}

prepare: ${ISO_WINDOWS10} ${ISO_VIRTIO} ${BIOS_ROM} ${VIDEO_ROM} ${DISK_IMG_FILE}
    sudo qemu-system-x86_64
        -enable-kvm -M pc -m 8G -cpu host
        -device vfat-pci,host=0x02.0,bus=pci.0,addr=0x4.0,multifunction=on,x-igd-gms=4,x-igd-opregion=on,rombar=0
        -device isa-debugcon,iobase=0x4f2,chardev=seabios
        -device stdio,fd=1,transparent,chardev=seabios
        -drive file=${DISK_IMG_FILE},cache=none,if=virtio,format=qcow2
        -device input-linux,id=mouse,evtdev=/dev/input/by-path/platform-18042-serio-1-event-mouse
        -device input-linux,id=keyb,evtdev=/dev/input/by-path/platform-18042-serio-0-event-kbd
        -device virtio-net-pci,netdev=eth0 -netdev user,id=eth0
        -device virtio-rgn-pci
        -rtc base=localtime
        -drive file=${ISO_VIRTIO},media=cdrom
        -drive file=${ISO_WINDOWS10},media=cdrom
        -device nec-usb-xhci -device usb-host,vendorid=0x04f2,productid=0xb67c
        -vga std -display gtk

snap-list:
    ls -lh ${DISK_IMG_FILE}
    qemu-img snapshot -l ${DISK_IMG_FILE}

clean:

clean_all: clean
    rm -f ${DISK_IMG_FILE}
```
Summary: Progress so far

What do we have?

- Successfully running Fedora 30 headless on Lenovo Carbon X1 7th gen
- Virtualized Windows 10 Pro boots in 3-4 seconds after starting QEMU: Snappy Operation!
- Extraction of VGA Bios from UEFI BIOS image and usage in QEMU virtualized Windows 10 Pro boot for initializing the graphics chip set
  - Virtualized Windows OS uses native hardware accelerated Intel Graphics Drivers: fast as hell!
- TrackPoint and Keyboard Support
- Hardening of BIOS Security Settings
- Scan of Mainboard PCB
- Decapping & chip-die-imaging of critical/suspicious chips – and initial threat modelling

What are we working on?

- Implement secure & measured boot process with static file system (dm_verity)
- Implement signed atomic updates for hardened Linux
- Implement Secure Crypto Token with JAVA JCOP3 card: YubiKey is unfortunately out due to lack of Secure Messaging support.
- Integrate LUKS2 Disk Encryption with Clevis / Dracut to support the Secure Token, Password and dTPM2 at the same time
- Add graphical interface to password entry (currently blind password entry)
- Plausible-deniability OS: Two windows OS alternatives: One secure and the other n
- Add hotkeys for snapshots and reverting to previous versions
- Touchpad support
- Power Management & Suspend-to-Disk
- USBGuard security for USB Webcam
- Network filtering & VPN
- WIFI Configuration from Windows
Questions?
Please ask!

@FoolsDelight or milosch@meriac.com
Slides @ www.meriac.com/hitb2019

... and of course, we’re hiring:
xen1thLabs
www.darkmatter.ae/xen1thlabs/
... please contact me!