



MALWARE – THE PHYSICAL IMPACT

Emily SJ

2022



Contents

Introduction	2
What is malware?	2
What will this report cover?	2
Viruses	3
What are viruses and how do they cause damage?	3
Chernobyl Virus	3
Rootkits	5
What are Rootkits?	5
Hardware rootkit	5
Trojans	6
What are trojans?	6
Functional and parametric hardware trojans	6
Worms	7
What are worms?	7
Stuxnet worm	7
PDoS	8
What are PDoS attacks?	8
PDoS attack approaches	8
BrickerBot	8
malwareIdentifier code	9
PowerShell script	9
Summary	11
Key discoveries	11
Resources used:	12

Introduction

What is malware?

Malware is software developed and used by cybercriminals to steal data and damage or destroy computers and computer systems.

What will this report cover?

The general focus of academic research into malware is its impact on other software, file systems, etc. This report will instead examine the lesser explored impacts of malware – their effect on hardware, identify the hardware components most impacted by successful malware execution, the cause of this impact and present a PowerShell script which, when run on a device, obtains key hardware components' temperature and compares it to their 'normal' operational temperatures as well as BIOS (The basic input/output system which is stored in memory and runs when the device boots) information, and indicates to the user whether it is likely a malware attack has or is occurring.

Viruses

What are viruses and how do they cause damage?

Viruses, malicious software attached to a legitimate file that sits dormant on a system until said file is executed, are the most common form of malware that's execution leads to the damage of hardware.

This damage is predominantly caused by the excess heat generation that occurs as a result of malicious virus code manipulating components to run at speeds and for durations beyond which they were intended. Consider the CPU, this is a common piece of hardware targeted by viruses, namely power viruses, which execute machine code to overclock the CPU. Overclocking is such that the CPU is run at a greater speed than initially set by the processor. Internally this comprises the CPU transistors being turned on and off an increased number of times per second. This more frequent transistor action increases the heat generated by the CPU as more resistance is created against current attempting to flow through the chip. This increased heat can lead to the thermal paste between the CPU and CPU cooler (which facilitates heat transfer from the processor) drying out. Without this, the heat is not sufficiently dissipated resulting in CPU temperature surpassing its maximum threshold (for most CPU's this sits around 90-95 degrees Celsius) and as a result, chip components and even the surrounding motherboard are broken (in some cases beyond repair) as the expansion, etc caused by the heat, can lead to the boards circuitry being damaged; current being unable to reach attached components and in some cases, the BIOS being unable to run. The only solution in this instance would be to replace all impacted hardware.

It should be noted that complete chip destruction due to heat is less likely with modern CPU's as a result of inbuilt thermal protection methods such as the chip being coded to shut down if worked too hard, however, it is still a risk and rapidly increasing CPU temperature a key indicator of a virus being executed.

Damage caused by CPU overclocking is not, as previously conveyed, limited to the CPU. The CPU receives its required voltage (normally between 0.9 and 1.3 volts) via the Voltage regulator module (VRM). The VRM takes the voltage output by the devices power supply (for example a battery) and converts it to the voltage required by the varying motherboard components to operate. During overclocking, the CPU requires a higher voltage to support the increased transistor action. When handling this, the VRM can overheat as it will need to draw more power from the power supply. Whilst the VRM can dissipate heat generated by current passing through its transistors; the increased current required in this instance leads to more heat being produced than can be dissipated. The subsequent damage to transistors which are sensitive to heat, will inhibit the VRM's function and as a result the correct voltage will not be delivered to components like the GPU, damaging them as they will either be fried by voltages higher than their circuits can withstand (in the example of the CPU, voltages over 1.5 volts would have this effect) or be unable to function stably using voltages that are too low.

Chernobyl Virus

The Chernobyl virus was the first virus ever discovered to have the capability of impacting computer hardware. Receiving its name following its first execution on the 13th anniversary of the Chernobyl incident, the virus affected Windows 9x-based Operating systems and spread using the portable executable (PE) files that contained the object code, etc used by the OS to manage

executables, as the viruses malicious code was split and inserted into the gaps in the file headers.

The virus comprised 2 key payloads, the first overwrote the first megabyte of data on the hard drive which housed the partition table, replacing it with zero's. The partition table is a crucial piece of information as it describes how the disk is partitioned, what is stored where and as a result is queried when trying to locate specific information such as files required for the booting process. It's overwriting thus usually lead to the windows blue screen that indicated an error and overall prevented the device from booting as the files required to complete the boot process could not be located or thus executed.

The second payload attempted to write to the Flash BIOS(the devices BIOS stored on a flash memory chip rather than a ROM chip), replacing key boot-time code with random content. If this was successful it would prevent the device from booting at all as the instructions detailed in the boot-time code, which needed to be executed in sequence to ensure hardware devices were activated in the correct order and corresponding software was executed in conjunction, were no longer present or thus runnable and as a result the device start up process would not proceed from the initial reference of the BIOS code.

Overall, this virus did not physically damage hardware but prevented it from functioning, rendering it useless as the device start up process was (due to the memory manipulations by the virus) not proceeding far enough without error to reach the code and programs that needed to be executed to trigger certain hardware components to run.

Rootkits

What are Rootkits?

Rootkits are often a collection of software that provides an attacker remote access and control of a device. Their detection is hard but not impossible using methods such as signature scanning. Rootkits are not usually solely responsible for damage to a device and its hardware, but instead used in conjunction with other forms of malware to achieve a certain end goal - in fact Rootkits are often used to help mask an attackers malicious actions on a device, being installed right after they gain access to make subsequent actions harder to discover. Hardware rootkits however, do impact the function of hardware.

Hardware rootkit

Similar to the Chernobyl virus, hardware rootkits impact computer hard drives or BIOS. The remote admin tools that the rootkit provides enables data on the disk to be altered as needed. Initial device infection, before the rootkit can be installed and or used, is achieved by running an executable from the OS. This is usually an infected file on the hard disk present following a user falling victim to a phishing email, etc or an attackers physical interaction with the system.

Whilst, with the remote admin access a rootkit provides, an attacker could arguably do anything, a common course of action is for them to modify the master boot record (MBR) which is located in sector 0 of the hard disk and identifies where the system OS is located so that it can be booted into RAM or main memory and then used, such that the OS cannot be detected. The consequence of this is an *'operating system not found'* error being displayed when a user attempts to turn on the device. It can even cause the computer not to boot.

Although physical damage will not be caused, this can render the physical hardware useless and in some cases unable to be used again as removal of this kind of root kit is very difficult, often requiring hardware to be replaced or specialised equipment to be used. To give some examples of how hardware is impacted, as UEFI/BIOS (software that links hardware to OS) will be unable to locate boot files and put them in RAM, the CPU will then not be started, RAM won't be occupied, etc and thus their functionality will not be fulfilled.

Trojans

What are trojans?

A trojan is a type of malicious code or software that appears to be legitimate but can take control of a device with the aim of damaging, disrupting, etc the device and its data. Most trojans are harmless to the hardware of a device, however, hardware trojans which can be inserted anywhere in a microchip e.g: processor, power supply, etc and maliciously modify circuitry of integrated circuits through physical manipulation of the victim device; can cause vast damage to a device, destroying parts of or entire chips.

It should be noted that as of the time of this report, August 2022, there is no concrete evidence of a successful hardware trojan execution, although that is not to say one has not occurred. Current detection methods rely on post-manufacturing testing to look for areas where development processes, etc have not been adhered to; attackers know this and can alter the ways they install trojans to fit these processes.

Functional and parametric hardware trojans

There are 2 types of hardware Trojans, functional and parametric. Functional hardware trojans entail transistors or gates being added to the chip design. Transistors are gates for electronic signals, these can facilitate or block flow of electricity leading to resistance occurring and heat being generated. Thus the more transistors added to a chip, the more heat that is generated. As transistors have a thermal limit above which they can break down and damage the circuit board, this additional heat generation would lead to significant damage to the circuit board inhibiting the overall device from functioning as current and data flow would be disrupted.

In contrast, parametric hardware trojans entail the modification of circuits such as the thinning of wires, weakening of flip flops or transistors, subjecting the chip to radiation, etc. A thin wire poses a greater resistance to current flow as there are fewer paths for the electrons to take without colliding than in a thick wire which again, as seen in the functional hardware trojan, would lead to extra heat production and thus circuit board damage. Use of radiation would instead damage the electrical components of the chip inhibiting it from being able to conduct and thus disrupting transmission of key data such as the clock signal which keeps hardware components in time with one another and enables contents of memory locations to be divulged.

Overall, no matter which form of hardware trojan was being used, it's presence would almost certainly guarantee complete damage of a devices circuit board and thus the device ceasing to function.

Worms

What are worms?

Worms are a type of malware that's primary function is to self-replicate and infect other computers while remaining active on infected systems issuing harmful commands, etc.

Stuxnet worm

Whilst the Stuxnet worm did not cause damage to hardware of the device it infected it's presence did lead to damage to other systems hardware and mechanical components and thus it is a prime example of how malware can manipulate and damage physical systems.

The Stuxnet worm was used in an attack on Iran's nuclear program. It caused centrifuges that span uranium into weapons-grade material to malfunction, they were made to spin excessively and the consequent strain on the hardware caused 1000 computers to implode. The worm spread through windows computers predominantly being transmitted using USB sticks. It targeted PLC's (programmable logic controllers – specialised computers used in industry to control manufacturing processes like robotic devices. They have one piece of code which they execute repeatedly until either an error occurs or there is intervention from an admin) by looking for devices running Siemens step 7 software which PLC's ran for automating and monitoring equipment. When a device running this software was found malware blocks were installed into the PLC monitors. These were used to change the systems frequency and in turn affect the operation of motors by changing their operational speed, in this case changing centrifuge operating speed such that there were spun too quickly and for too long leading to the damage and destruction of delicate equipment. It also contained a rootkit that hid the worm from monitoring systems such that whilst the destruction was taking place, the PLC would continue to tell the controller computer that everything was working fine, inhibiting diagnosis until it was too late thus maximising damage caused and guaranteeing attack success.

PDoS

What are PDoS attacks?

Permanent denial of service attacks achieve denial of a service through hardware sabotage. Security flaws in network hardware are exploited by an attacker who then replaces the devices firmware (software programmed into ROM – normally a networking device) with a modified version inhibiting the device from functioning correctly and needing to be replaced.

PDoS attack approaches

There are multiple methods through which PDoS attacks are achieved; one, as seen with the power virus, uses high voltage to damage hardware. An attacker plugs a modified USB into the intended victim device. When inserted, the USB releases a 220v negative electric charge frying motherboard components like circuits as its vastly above the voltage such components can withstand. The motherboards can then no longer conduct and thus itself and any hardware plugged in has to be replaced permanently preventing that hardware from being able to deliver its function.

An alternative approach is 'Phlashing'. This entails devices with embedded systems being targeted and known vulnerabilities in their firmware being exploited. This in turn leads to the firmware failing, inhibiting the device from being able to function such that it has to be replaced.

BrickerBot

This PDoS attack targeted vulnerabilities in IoT devices to render them permanently useless. Discovered in April 2017, the BrickerBot used brute force dictionary attacks (a phlashing approach) to obtain device's with open telnet connections, telnet password. This enabled the attacker to gain remote access to the device and then execute a series of malicious Linux commands which, to give a few examples, corrupted device storage such as multi media cards, deleted all files and inhibited connection to the internet and other devices.

As previously seen in the example of rootkits, this manipulation of memory and complete deletion of files would prevent the device from being able to boot, as executables vital to the boot process would no longer be locatable as file storage records had been corrupted so reference to storage location could not be obtained, or because the files had been part of the collection deleted. Additionally, although disconnection from the internet may not have stopped the IoT device from functioning, inability to connect to other devices would have, as connectivity of IoT devices is required for certain functions to be completed. Consider an IoT lightbulb activated when motion is detected. Were it to be targeted by the BrickerBot, the likely impact would be that the bulb would no longer turn on when something moved. This is because, IoT bulbs do not have inbuilt motion sensors, instead they connect to a motion sensor on their network which, when it detects motion, sends a signal to the light bulb triggering it to turn on. The inability to connect to the motion sensor following the BrickerBot attack would mean the bulb no longer receive the motion detection prompts and thus would remain in an off state 'believing' that no motion was occurring. Overall, the only method of resolution after falling victim to the BrickerBot would be to completely replace the device thus again posing another example of malware permanently preventing a piece of hardware from operating, not through physical damage, but through manipulation of the code and files that are vital to its function.

malwareIdentifier code

PowerShell script

The below PowerShell script, which can be executed in the PowerShell ISE when run as administrator, could be used to help detect the presence of malware on a device and thus protect hardware components from damage.

The code obtains information pertaining to two of the key signs of malware presence on a device as discovered in the above research. Firstly, operational temperature of motherboard components is considered as the user devices CPU temperature is obtained. The below section of the overall 'malwareIdentification' script demonstrates the way in which I used a windows management instance object to obtain the temperature in the thermal zone that the CPU resides in. For context, the motherboard and other hardware components of a device are split into different 'thermal zones', these all have dedicated thermometers which monitor the temperature in that area which enable us to check, as in this instance, whether the temperature in that section fits the requirements of its components.

This method returns the desired temperature in 'non-standard' units, thus I convert the temperature into Celsius. The temperature in Celsius is then evaluated and depending on whether it falls within or below the normal CPU temperature range, is slightly high or is close to the temperature (over 90 degrees) at which permanent damage to hardware would be caused, a different error message with a suggested course of action is displayed.

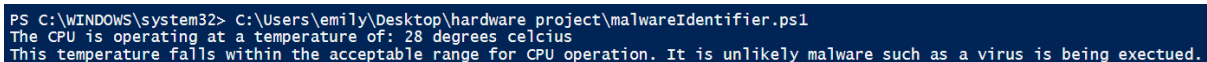
```
#get cpu temperature
$data = Get-WmiObject MSAcpi_ThermalZoneTemperature -Namespace "root/wmi"
$kelvin=@()
$kelvin = ($data.CurrentTemperature) / 10

#convert to celcius
$cels = $kelvin - 273

#display CPU temperature regardless of whether its 'normal' or too high
Write-Host "The CPU is operating at a temperature of: $cels degrees celcius"

#different error messages are displayed based on the extent of damage persistant operation at this temp would cause.
if($cels > 85)
{
    Write-Host "CPU temperature is extermely high. If it continues operating at this temperature much longer
    permenant damage to hardware will occur. It is likely a malicious agents actions are causing this issue so either turn
    off your device or take steps to review running processes and turn off those that are unfamiliar"
}
elseif($cels > 70)
{
    Write-Host "This is above the normal temperature range for a CPU (40 - 70 degrees celcius) this could indicate
    processor overclocking is occurring, investigate what is currently running on your device to ensure a malicious
    executable is not being run"
}
else
{
    Write-Host "This temperature falls within the acceptable range for CPU operation. It is unlikely malware such as a
    virus is being execteded."
}
```

Temeptrature check section of the malwareIdentifier script. The full script can be obtained from this reports supporting documentation



```
PS C:\WINDOWS\system32> C:\Users\emily\Desktop\hardware_project\malwareIdentifier.ps1
The CPU is operating at a temperature of: 28 degrees celcius
This temperature falls within the acceptable range for CPU operation. It is unlikely malware such as a virus is being execteded.
```

The above image depicts the result of this scripts execution on my personal device. As you can see, my CPU is operating at 28 degrees Celsius, this falls within, in fact below, the standard operating temperature for a CPU as identified in my prior research to be between 40 and 70

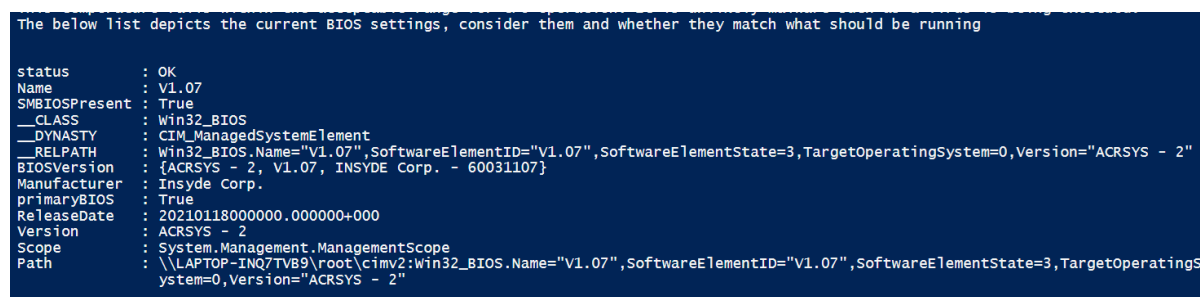
degrees. This indicates, as conveyed by the displayed message, that overclocking is not taking place and that it is unlikely a malicious agent is operating on my device.

Secondly, device BIOS information is obtained and displayed. As seen in the impacts of hardware rootkits for example, another predominant method attackers use to inhibit hardware function is alteration of the BIOS. When BIOS information has been changed it can be very difficult, if not impossible, to obtain through PowerShell commands, the details of the BIOS in its prior state. In light of this, I have chosen to obtain and display key characteristics of the users BIOS in the form it currently takes. The user should then examine this, look at storage locations, versions, and manufacturers, to look for clues that it has been changed by a non-legitimate party. The code I used to achieve this is shown below:

```
#check BIOS information
Write-Host "The below list depicts the current BIOS settings, consider them and whether they match what should be running"

#get key BIOS details - enables comparison with what you know should be installed.
Get-WmiObject -Class Win32_Bios | Format-List -Property status,Name, SMBIOSPresent, __CLASS,__SUPERCLASS,__DYNASTY,__RELPATH,__DERIVATION,__NAMESPACE,__PATH,BIOSVersion,Manufacturer,primaryBIOS,ReleaseDate,Version,Scope,Path
```

As you can see, I again use a windows management instance object to obtain the information. The Win32_Bios class returns a vast quantity of information, however not all of this would be relevant or understandable to a user looking to identify unexpected changes, thus, using the Format-List '-property' attribute I reduced volume of information returned by filtering to key characteristics.

A screenshot of a PowerShell terminal window with a dark blue background and white text. The first line is a prompt: "The below list depicts the current BIOS settings, consider them and whether they match what should be running". Below this, a list of BIOS properties is displayed in a key-value format. The properties include status (OK), Name (V1.07), SMBIOSPresent (True), __CLASS (Win32_BIOS), __DYNASTY (CIM_ManagedSystemElement), __RELPATH (Win32_BIOS.Name="V1.07", SoftwareElementID="V1.07", SoftwareElementState=3, TargetOperatingSystem=0, Version="ACRSYS - 2"), BIOSVersion ({ACRSYS - 2, V1.07, INSYDE Corp. - 60031107}), Manufacturer (Insyde Corp.), primaryBIOS (True), ReleaseDate (20210118000000.000000+000), Version (ACRSYS - 2), Scope (System.Management.ManagementScope), and Path (\\LAPTOP-INQ7TVB9\\root\\cimv2:Win32_BIOS.Name="V1.07", SoftwareElementID="V1.07", SoftwareElementState=3, TargetOperatingSystem=0, Version="ACRSYS - 2").

```
status      : OK
Name        : V1.07
SMBIOSPresent : True
__CLASS     : Win32_BIOS
__DYNASTY   : CIM_ManagedSystemElement
__RELPATH   : Win32_BIOS.Name="V1.07",SoftwareElementID="V1.07",SoftwareElementState=3,TargetOperatingSystem=0,Version="ACRSYS - 2"
BIOSVersion : {ACRSYS - 2, V1.07, INSYDE Corp. - 60031107}
Manufacturer : Insyde Corp.
primaryBIOS : True
ReleaseDate : 20210118000000.000000+000
Version     : ACRSYS - 2
Scope       : System.Management.ManagementScope
Path        : \\LAPTOP-INQ7TVB9\\root\\cimv2:Win32_BIOS.Name="V1.07",SoftwareElementID="V1.07",SoftwareElementState=3,TargetOperatingSystem=0,Version="ACRSYS - 2"
```

The above image demonstrates the output of this section of the overall malwareIdentifier code on my personal device. The release date corresponds to when I purchased the laptop (indicating that the BIOS has not been changed since), location of where the BIOS is, is as expected, etc all of which indicate that a rootkit or other piece of BIOS altering malware has not been used on my device.

Whilst I appreciate the BIOS information acquisition section of this script does require user review and input instead of a user being presented with a standard 'yes or no' regarding whether or not it is likely their BIOS has been manipulated, I still believe it is key information to obtain and display as it could be used as a basis of further research for a course of action or to indicate to the wider community a new potential threat.

Summary

Key discoveries

Overall, the hardware components most targeted and affected by malware are the CPU and hard drives. The predominant causes of this damage are: excess heat production caused by malware stimulating a component to operate at speeds beyond its capacity leading to transistors and other vital circuit components being destroyed and manipulation of device BIOS and memory. As a result component temperatures, specifically that of the CPU, should be carefully monitored and device BIOS information regularly obtained and reviewed for unexpected changes, in a continued effort to detect malware and prevent damage to hardware.

Resources used:

- <https://www.cisco.com/c/en/us/products/security/advanced-malware-protection/what-is-malware.html>
- <https://uk.norton.com/internetsecurity-malware-what-is-a-computer-virus.html> 13/07/2022
- <https://motherboardscan.com/guide/does-motherboard-affect-performance/>
- <https://forums.tomshardware.com/threads/will-overclocking-cpu-fsb-cause-increased-vrm-mobo-temp.1763975/>
- <https://forums.tomshardware.com/threads/can-overclock-damage-my-motherboard.3277196/>
- <https://blogs.umass.edu/Techbytes/2017/02/08/cpu-overclocking-benefits-requirements-and-risks/#:~:text=When%20overclocking%2C%20what%20we're,shorter%20lifespan%20for%20the%20chip.>
- [https://en.wikipedia.org/wiki/CIH_\(computer_virus\)](https://en.wikipedia.org/wiki/CIH_(computer_virus))
- <https://www.techtarget.com/searchsecurity/definition/Chernobyl-virus>
- <https://www.google.com/search?q=how+did+the+Chernobyl+virus+damage+hardware&oq=how+did+the+Chernobyl+virus+damage+hardware&aqs=chrome..69i57j33i160.7073j0j7&sourceid=chrome&ie=UTF-8>
- <https://www.computerhope.com/issues/ch001119.htm>
- https://arbynthechief.fandom.com/wiki/Permanent_Denial_of_Service
- <https://blog.radware.com/security/2015/10/ddos-fire-forget-pdos-a-permanent-denial-of-service/>
- <https://www.trendmicro.com/vinfo/us/security/news/internet-of-things/brickerbot-malware-permanently-bricks-iot-devices>
- <https://www.trgdatacenters.com/what-is-a-permanent-dos-pdos-attack-and-can-i-stop-it/>
- <https://www.urtech.ca/2020/10/solved-what-is-a-permanent-denial-of-service-pdos-attack-in-simple-terms/>
- <https://us.norton.com/internetsecurity-malware-what-is-a-trojan.html#>
- <https://www.techdesignforums.com/practice/guides/hardware-trojan-security-countermeasures/>
- <https://www.minitool.com/data-recovery/pc-keeps-shutting-down-without-warning.html>
- <https://support.microsoft.com/en-us/topic/-operating-system-not-found-or-missing-operating-system-error-message-when-you-start-your-windows-xp-based-computer-e477fa16-69b3-343f-b293-d8fbe5fd4608>
- <https://www.acronis.com/en-gb/blog/posts/boot-sector-virus/>
- <https://us.norton.com/internetsecurity-malware-what-is-a-rootkit-and-how-to-stop-them.html#>
- <https://hps.org/publicinformation/ate/q11162.html>
- https://en.wikipedia.org/wiki/Hardware_Trojan#:~:text=If%20a%20Trojan%20is%20activated,is%20a%20very%20rare%20event.
- <https://www.secure-ic.com/blog/hardware-trojans/hardware-trojans/>
- <https://www.bbc.co.uk/bitesize/guides/z9sb2p3/revision/3#:~:text=The%20relationship%20between%20resistance%20and%20wire%20length%20is%20proportional%20.&text=The%20resistance%20of%20a%20thin,a%20wire%20is%20inversely%20proportional%20.>

- <https://www.techtarget.com/searchsecurity/definition/worm>
- <https://smartertermssp.com/can-malware-physically-damage-a-computer/>
- <https://nordvpn.com/blog/stuxnet-virus/#:~:text=Stuxnet%20installs%20malware%20blocks%20in,the%20worm%20from%20monitoring%20systems.>
- <https://www.csoonline.com/article/3218104/what-is-stuxnet-who-created-it-and-how-does-it-work.html>
- <https://www.trellix.com/en-us/security-awareness/ransomware/what-is-stuxnet.html>