Atharva Vilas Gorde. International Journal of Engineering Research and Applications
www.ijera.com
ISSN: 2248-9622, Vol. 11, Issue 11, (Series-I) November 2021, pp. 01-08

RESEARCH ARTICLE OPEN ACCESS

A Study of Stuxnet Malware: Simplification of Its Working Mechanism.

Atharva Vilas Gorde
Student, Department of Computer Engineering, Smt. Kashibai Navale College of Engineering, Pune-041, India.

ABSTRACT
The Internet has turned our world upside down. It has revolutionized all the things we do. It has been transformed from time to time and is now embedded in each and every aspect of our day-to-day life. Most of us are aware of its good cause but no one tends to look for the evil one. As we say the Internet has changed various things, it has changed the traditional way of battles too - Cyber war. A term which is gaining popularity in recent times. Cyber warfare refers to attacks which are deployed virtually in order to disrupt any country's stability; it might result in a drop of economy or even death of innocent lives. The Iranian Nuclear Facility at Natanz faced a cyber-breach which led to a blackout until emergency power systems kicked in. The result was what sounded like a slow-motion explosion as centrifuges crashed into each other. This paper is targeted to examine working and construction of the Stuxnet, which will include elaboration of various aspects of this targeted malware.

Keywords - Air-gapped, Centrifuges, Malware, Mechanism, Rootkit, SCADA, Stuxnet, Zero-day.

Date of Submission: 26-10-2021 Date of Acceptance: 10-11-2021

I. INTRODUCTION

In early 2010 a computer virus was discovered in thousands of the control systems which operate the factories, power plants and various nuclear reactors in the world. This virus was claimed to be 20 times more sophisticated than any malware ever reported, as it could halt oil pipelines, destroy water treatment plants and bring down the whole power grids. One of the four companies globally has been victims of this extortion caused by Stuxnet. Other victims of Stuxnet include 80% of the power companies of Mexico and over 60% of power companies of India.[1]

In 2010 Iran replaced more than 1000 centrifuges which counted 20% of the total centrifuges in the Fuel Enrichment Plant (FEP) at Natanz. Although mechanical failures were discussed as the cause for crashing these centrifuges, crashing off nearly one fifth of total centrifuges over a relatively short period of time raised suspicion and later it was undisclosed that it was the result of an infection of the Stuxnet malware. This malware was crafted in order to gain control and destroy the equipment while hiding its presence to make detection of the malware more difficult. It was most likely made to target Iran to destroy maximum possible centrifuges and set back Iran's progress in Nuclear Programmes. The attack was worked out by forcing a change in the centrifuge's rotation speed by first increasing it and then lowering it, the intention was to induce excessive vibrations or distortions, which would end up in bowling up the centrifuges.[2]

Meanwhile a computer security firm from Belarus got a clue as they received a request from one of their clients in Iran. Their machines were rebooting over and over again, even wiping all the hard drives and reinstalling the operating system was not working. The term Stuxnet was named here for the very first time, when the technicians tore down the whole operating system to find the root cause of this whole disturbance in the systems which was this new and unusual virus. Stuxnet was affecting computers all over the world and was spreading very fast. The cyber security experts knew that it wasn't any ordinary piece of code, as the size of the code base was about 500 kilobytes when compressed and about 1.2 megabytes when uncompressed, which was a large piece of code to go undetected. When analyst transferred Stuxnet in the state of the art test highly protected workstation designed for cyber security threat detection, it was observed that as soon as Stuxnet's files were copied to this computer, it was immediately infected without any user inputs also without triggering even a single alert by any of the threat detection programs which were pre-installed in it. It was very unusual as any software which is being installed over any computer needs Digital

Certificates which are checked against trusted manufacturers. Most of the viruses temper these certificates in order to trick the operating system into allowing it to install, usually these types of tempered certificates are caught by most of the Antivirus Programs. But in the case of Stuxnet, it didn't have any altered certificates; it had valid certificates which were stolen from the most used driver sources which we will discuss later. Makers of the Stuxnet made sure that windows will install Stuxnet without triggering any alert using these valid but stolen certificates. As more information about Stuxnet was revealed it added that as soon as the Stuxnet was installed it started searching for any sort of storage device be it a flash drive, USB stick, or any hard drive. At this moment the experts claimed that they discovered one of the rarest and most dangerously sophisticated kinds of software vulnerabilities. [3]

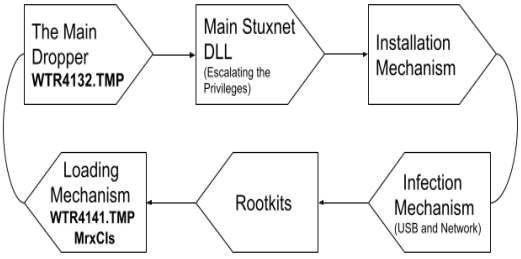
Stuxnet is an era of the new and highly sophisticated malwares. The Stuxnet worm was discovered in 2010 this piece of malware had surprised the computer experts due to its sophistication and use of all the four zero-day exploits.[4] This malware might have been planted using USB drive as this was the only technique to breach in any air gapped network which had great security as these networks were physically isolated from other networks.[5] This complex threat used four zero-day vulnerabilities in the windows OS which helped the virus to avoid being detected by the defensive antivirus programs which damaged the nuclear reactors in Natanz's nuclear facility by gaining access over its PLCs which controlled the machines there. That made it modify the control program which changes the behavior of the machine.[4]

Most of the devices used security software in order to white-list the processes which are being carried out. And these software white listed the process on the basis of the digital certificates so it was like a piece of cake for Stuxnet to trick them. This malware was targeted to perform the act so efficiently and quietly that no one would even notice. And we can say it outperformed it. In this paper we will be going through various aspects of this malware.

II. ATTACK SCENARIO

The following was the possible attack scenario:
The PLC and Industrial control systems (ICS) are generally programmed to function without any internet connection. So first of all the attackers might have conducted reconnaissance in order to gain the potential knowledge of the computing environment in the facility. Here they found out that there were 4 ways through which the rootkit distributed itself: first was by means of flash drives, through the network shares, through an RPC vulnerability and through the MS10-061 Print spooler vulnerability.[9] Then the attackers tested their code on the replica of this environment to ensure whether it is working and executing itself properly. Since the malware had valid stolen digital certificates, someone might have physically entered the premises of these two companies namely JMicron and Realtek to get those certificates. Now in order to infect the network the Stuxnet had to be introduced to the target environment by infecting a willing or unknowingly third party. The original infection might have been introduced to target by USB. Once the target was infected Stuxnet would first try to spread on the LAN using the known 4 zero-days. And would then follow the cycle which would had lead it to escalate its own privileges and gain the control over the target.[6]

III. STUXNET'S WORKING MECHANISM



3.1. THE MAIN DROPPER {WTR<RND>.TMP}

The rootkit connects as driver filter and thereby gains control over the infected computer's files and hides them under name WTR<rnd>.TMP, in WTR<rnd>.TMP rnd is a random 4 digit number which might be WTR<4132>.TMP or WTR<4141>.TMP. to show its authenticity it has a Realtek Semiconductor Corp digital signature.[8] It begins its execution by searching for ".stub" section, which contains the main Stuxnet DLL files like it's functions, mechanisms, files and rootkits. This section might be considered as the hub of all the processes which are implemented by the Stuxnet in order to gain the privileges.

Once the section's location is validated it avoids the security software, to do so the malware loads a module in the following way:

1. Firstly it allocates the buffer memory for the calling process of the module which is to be loaded.
2. Then it patches the Ntdll.dll system library: which consists of the following APIs:

a. ZwMapViewOfSection

b. ZwCreateSection
c. ZwOpenFile
d. ZwClose
e. ZwQueryAttributesFile
f. ZwQuerySection

Now with the ZwOpenFile it makes sure that the .stub section file is readable. These patches make LoadLibraryA load a DLL file from a different location of the memory instead of the harddisk. The DLLName for this LoadLibraryA is like KERNEL32.DLL.ASLR.XXXX to load the Main DLL File it calls function #15 at the end.[10]

3.2. MAIN STUXNET DLL

3.2.1. ESCALATION OF THE PRIVILEGES AND INJECTION OF THE NEW PROCESS

When the DLL initiates its execution it tries to confirm the environment and the configuration data. If things aren't in place it will exit and re-initiate the process from beginning. By things I refer to the configuration data and the admin rights. If it is not running on the admin level it will use one of the two zero-day vulnerabilities to escalate the privileges to the administration level.

CVE-2010-2743(MS-10-073) - WIN32K.sys Keyboard Layout Vulnerability.

CVE-xxxx-xxxx(MS-xx-xxx) - Windows Task Scheduler Vulnerability.

As these two vulnerabilities help to gain admin access they also make some other checks like checking whether the system is of 64 bits or 32 bits. Once the environment is prepared to get the infection from the Stuxnet, it injects itself to another process which identifies the security software or the Antivirus programs installed in the machine. If there is not antivirus program on the machine it goes with ”Isass.exe”.[4] Below are injection targets in correspondence to their security product validated by Symantec.[6]

Process Injection

Security Product Installed	Injection target
KAV v1 to v7	LSASS.EXE
KAV v8 to v9	KAV Process
McAfee	Winlogon.exe
AntiVir	Lsass.exe
BitDefender	Lsass.exe
ETrust v5 to v6	Fails to Inject
ETrust (Other)	Lsass.exe
F-Secure	Lsass.exe
Symantec	Lsass.exe
ESET NOD32	Lsass.exe
Trend PC Cillin	Trend Process

What makes Stuxnet more special is rather than searching for the task manager to inject itself it creates a new process using the CreateProcess. Now it unloads, for example the lsaass.exe from its memory and loads it to another Portable Executable (PE) file from its DLL resources in the same place where it unloaded the isass.exe module previously. Stuxnet makes some modifications by adding a new section named “.verif” before loading this newly created PE file. This .verif section’s size is equal to that of previously unloaded modules and then Stuxnet writes a “jmp” instruction to the initial point of this PE file. Here comes the role of previously allocated buffers, in this step the Stuxnet copies the .stub section from the main DLL to the memory of the infected process and then it writes the .bin section the pointer to its memory buffer. To wrap up thing the Stuxnet resumes the main thread if the infection process and reloads the Main Stuxnet DLL and calls the next function.[4]

3.2.2. INSTALLATION OF STUXNET INTO THE INFECTED MACHINE

As usual it again checks for the surroundings and to begin the installation it validates whether there is a value with this name “NTVDM TRACE” in

SOFTWARE\Microsoft\Windows\Current Version\MS-DOS Emulation

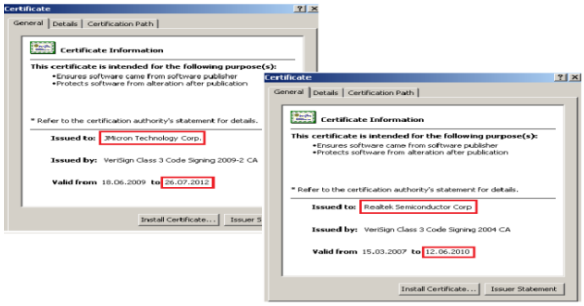
The value seems a bit arbitrary “19790509”; as it appears to be a date- May 9th,1979. Which might be a possibility. After checking this value Stuxnet installs itself by writing 6 files in Windows directory of which
4 are encrypted files

C:\WINDOWS\inf\oem7A.PNF
C:\WINDOWS\inf\oem6C.PNF
C:\WINDOWS\inf\mdmcpq3.PNF
C:\WINDOWS\inf\mdmeric3.PNF

and 2 are device drivers,

C:\WINDOWS\system32\Drivers\mrxnet.sys
C:\WINDOWS\system32\Drivers\mrxcls.sys[4]

Here comes another catch of the Stuxnet. The drivers mentioned above were issued for Realtek Semiconductor Corp. and JMicron Technology Corp. by VeriSign and were suspected to be stolen from the offices of Hsinchu Science Park, Taiwan. There is a possibility that these certificates might have been compromised by physical theft.[9] Since these certificates were valid even if any of the installed security software would have encountered the presence of Stuxnet, it would assume it to be a valid program.



After installing the drivers, Stuxnet makes sure that it will load in the very beginning of the boot that is before most of the windows applications every time. Now by calling ZwLoadDriver it loads the mrxnet just after adjusting the privileges and then adds SeLoasDriverPrivilege. So to avoid being obligated by the firewall, it modifies the windows defenders values in the below key:

SOFTWARE\Microsoft\ Windows Defender\Real-Time Protection

The changed values being: EnableUnknownPrompts, EnableKnownGoodPrompts, ServicesAndDriversAgent and sets them to zero and disables the firewall for itself. Here we have completed the installation mechanism of the Stuxnet. Now we will see how the Stuxnet spreads itself in the network.

3.3. SPREADING MECHANISM OF STUXNET

3.3.1. THE INFECTION CAUSED DUE TO USB DRIVES

As discussed in the attack scenario there were high chances that infection occurred due to USB by means of any willing or unwilling person who perhaps had access to these computers. Once the Stuxnet detects the new drive in the computer it writes 6 files into the flash memory drive: Copy of Shortcut to.lnk, Copy of Copy of Shortcut to.lnk, Copy of Copy of Copy of Shortcut to.lnk, Copy of Copy of Copy of Copy of Shortcut to.lnk And 2 executable files (DLL files): ~WTR4141.tmp and ~WTR4132.tmp. The shortcuts mentioned use the Windows Shell vulnerability which is:

CVE-2010-2568(MS-10-046) -Windows Shell LNK Vulnerability [9]

This vulnerability was created as windows used to load the icons for LNK files in a sort of wrong way. These shortcuts were special for an unknown types of file called CPL files i.e. Control Panel Item file by which we can create a shortcut identical to these shortcuts. It is quite complex to differentiate these malformed shortcuts. In these shortcuts the section begins with the process id of the control panel and then transforms to any other process id until it makes an item containing path and file name of Stuxnet DLL i.e. ~WTR4141.tmp. We have 4 shortcut files as every file contains a different form of path leading to wtr4141.tmp file in order to ensure the compatibility with all OS which had this vulnerability. These paths were as follows:[4][9]

For Windows7:
\\.\STORAGE#Volume#_??_USBSTOR#Disk&Ven_____USB&Prod_FLASH_DRIVE&Rev_#12345000100000000173&0#{53f56307-b6bf-11d0-94f2-1300a0c91efb8b}#{53f5630d-b6bf-11d0-94f2-00a0c91efb8b}\~WTR4141.tmp

For Windows Vista:
\\.\STORAGE#Volume#1&19f7e59c&0&_??_USBSTOR#Disk&Ven_____USB&Prod_FLASH_DRIVE&Rev_#123450001000000000173&0#{53f56307-b6bf-11d0-94f2-00a0c91efb8b}#{53f5630d-b6bf-11d0-94f2-00a0c91efb8b}\~WTR4141.tmp

For Windows XP, Windows Server 2003 and Windows 2000:
\\.\STORAGE#RemovableMedia#8&1c5235dc&0&RM#{53f5630d-b6bf11d0-94f2-00a0c91efb8b}\~WTR4141.tmp

For Windows XP, Windows Server 2003 and Windows 2000:
\\.\STORAGE#RemovableMedia#7&1c5235dc&0&RM#{53f5630d-b6bf11d0-94f2-00a0c91efb8b}\~WTR4141.tmp

These paths make sure that the code of Stuxnet is executed by explorer which afterwards calls API named "Shell32.LoadCPLModule" to load icons for this shortcut and calls previously mentioned LoadLibraryA API which helps in the execution of wtr4141.tmp.[4]

3.3.2. SPREADING VIA NETWORK

While spreading via network it uses one of the two vulnerabilities:
CVE-2008-4250(MS-08-067) –Windows Server Service NetPathCanonicalize() Vulnerability
CVE-2010-2729(MS-10-061) –Windows Print Spooler Service Vulnerability

But here the first vulnerability is not a zero-day as it was used by a fast spreading worm called Conficker in 2008.[7]

The second vulnerability was a zero-day one which wasn't used till date. This vulnerability allowed any foreign user account to establish communication to a machine which is connected to a shared printer and writes system directories in it.

This resulted in access to ReadFile and WriteFile APIs which we can copy in the target machine. In case of the Stuxnet it copied two files into the target computer:

Windows\System32\winsta.exe
- Stuxnet Dropper.

Windows\System32\wbem\mof\sysnullevnt.mof - Managed Object Format file.[4]

3.4. UPDATING AND REMOTE COMMUNICATION PROTOCOL

3.4.1. UPDATING STUXNET VIA INTERNET

Stuxnet used to make sure whether it is connected to the Internet by checking two websites www.windowsupdate.com and www.msn.com. If it failed to establish connection with one of these two sites it would stop sending the data. Stuxnet updated itself with the help of two fake websites by establishing HTTP connection with them, those websites were:

www.mypremierfutbol.com
www.todaysfutbol.com

When the Stuxnet confirmed the connection with the sites it sent the request to the remote server for example: http://www.mypremierfutbol.com/index,php?data= data_to_send where data_to_send was encrypted and encoded message which used the encryption algorithm whose key length was 31 bytes.[9] This data comprises the information regarding IP, the Adaptor name and some other related data of the infected machine and itself.

3.4.2. UPDATING VIA P2P CONNECTION

After infection the Stuxnet creates the RPC server to track any connections which come from any PC which is connected to the same or interconnected network. First it sends the Stuxnet version of the RPC server which is stated as Function 0, again if it is newer then it calls Function 1 which creates the RPC server's DLL files copy and sends it to the Stuxnet client. Once the receiver receives either the updated Stuxnet or the copy of DLL files it injects the chosen process and begins the Installation. By chance if the condition is reversed i.e. if the RPC has an older version then the client will call function 4 which will prepare a copy of the newer version of the Stuxnet file and will send it to the RPC server in order to install it. In this way Stuxnet updates itself in the networks without Internet connectivity. This way is implemented to spread the Stuxnet to all the indirectly connected computers on the grid.

3.5. ROOTKITS

3.5.1. USER-MODE ROOTKIT

Once the DLL file is loaded by the LNK vulnerability Stuxnet does two things first it loads as Main Stuxnet Dropper and secondly it works as a user mode rootkit in order to hide the files of Stuxnet in the flash memory. It injects the code to a module by either loading itself to an existing process or by injecting into a new process. In order to inject in the current process it allocates memory buffer code in order to dispatch calls to hook functions, then to transfer the control to new function it overwrites some data in the MZ header of the image and then hooks the original function to the overwritten data by overwriting its bodies.[9] All these steps make sure that the non-existing library is legitimately added to the ongoing process and those hook function have a crucial role, as they allow the malware to load the module as if it pre-existed. Basically these functions call the original functions and modify their output to hide the malwares files so also they check if they contain .LNK file with the specific size (4171 bytes) or a file named ~WTRxyzw.TMP where the x+y+z+w=10.[4] But as these DLLs are not stored on the hard drive some software may detect it and warn users as when malware tries to load the real DLL there is no file on the mapped location. So there is another way to deal with the situation by directly injecting it to a whole new process. To get this going Stuxnet first creates host process and then it replaces the images of the newly created process with the module which is to be executed in addition to this it adds up the supplementary code that will load the module and most importantly call the specified parameters which are based on the type of security software used. Generally it chooses the name of an executable image whose name is present in the list of the software and meets certain criteria.[9] This rootkit is used for once only as after its purpose is served Stuxnet installs another rootkit called as “MRxNet” which is a Kernel-mode rootkit.

3.5.2. KERNEL-MODE ROOTKIT

The previously created temporary files which are stored in the flash memory are needed to be covered so that’s the real cause of MRxNet. This prevents any user to notice the Stuxnet in any removable drive for if their drive was unintentionally infected or not. The code can be obtained by reversing this driver in C++. As the driver file contains the digitally signed certificates from Realtek hence it was never obligated by any software.[6] This driver scans for the filesystem driver objects and instead of modifying the address in the import table it adds to the driver chain of

these drivers : \\FileSystem\\ntfs, \\\\FileSystem\\fastfat, \\FileSystem\\cdfs. These are the drivers who are responsible for all the file transfers in the system by adding itself to this driver chain MRxNet makes sure that it will receive the request before the actual drivers receive any of the requests for transfer or process. By receiving prior requests the MRxNet is now able to modify the the input of these drivers and by this method it hides a directory named: {58763ECF-8AC3-4a5f-9430-1A310CE4BE0A} and after doing this it makes sure that to wipe the traces of modification it deletes the input (ISP) request to the actual drivers. The intention behind this process is to modify the output by adding the request to the IOCompleationRoutine.[4]

3.6. LOADING MECHANISM

3.6.1. ~WTR4141.TMP

The mechanism of this executable DLL is discussed earlier in the paper.

3.6.2. MRxCLs LOADER DRIVER

Various analyses have led to the conclusion that MRxCLs driver had different characteristics when compared to the other parts of the Stuxnet. What raised suspicion was the MRxCLs driver form Stuxnet was spreading in mid-2010 which was created on 2nd January 2009 but was signed on 26 January 2010 and that was the time when the other resources were built and signed. It seemed like the signing had to wait for the stolen certificates from Realtek and JMicron. What made it really distinct was that it was written in C++(where global variables were avoided), the Stuxnet's kernel mode driver was specially crafted to churn complete task of MRxCLs by avoiding the debug directory which also implied that there was no pathname for program database (PDB) file. The most notable thing was MRxCLs was written with far more generality that it had all the arbitrary data of the configuration which made it capable enough to name for each process and number of any DLL that was considered throughout the process.[10] It was complex enough to bypass all the security softwares and can be assumed to be a separate project as it was never modified when the versions of Stuxnet upgraded themselves. Now we will have a thorough look into how it really worked out.

MRxCLs reads "Data" value key as the parameter of the driver, the parameter is taken from the registry's key name:

HKEY_LOCAL_MACHINE\SYSTEM\ CurrentControlSet\Services\MRxCls

The "Data" is encrypted using 0xAE240682 as the key when decrypted, it has some system process and the filenames being used by Stuxnet. This "Data" specified the injection process' file location. The "Data" is so organized that all the elements in it have the access to the name of the infected process, later the DLL file injects the process and the key is used to decrypt the virus. Below is the Address and functions which contained the information written by the malware as an input "Data":

Component	Address	Function
services.exe	\SystemRoot\inf\oem7A.PNF	Main DLL and call to Export 1
S7tgtopx.exe	\SystemRoot\inf\oem7A.PNF	Scada Infection and call to Export 2
CCProjectMgr.exe	\SystemRoot\inf\oem7A.PNF	Call to Export 2
explorer.exe	\SystemRoot\inf\oem7m.PNF	Call to Export 2

If the flag was set to value 11 or 3 the Stuxnet was considered to be encrypted so then the driver had to decrypt it first by reading it.[10,4]
After adding the "Data" to pre-registered MRxCLs via the key:

SYSTEM\CurrentControlSet\Services\MRxCls

The values of parameters makes it load as a boot driver and makes sure that it will load before other services or drivers. Then the parameters from Data decrypts it and saves it into a tabular format. The next step is that it checks for InitSafeBootMode and KdDebuggerEnabled. Then it proceeds to create the IoCreateDevice API which finally creates a new driver called \Device\MRxCLsDvX. Now it adds some function to get details like version and APCs status without using GetProcAddress instead it uses the function which is called as: MmGetSystemRoutineAddress.to end the initialization it calls the function which will register every time of the process and which will keep look after a module if it is loaded in the memory excluding own files.

Now in order to inject in the kernel-mode, the process begins by checking for the "kernel32.dll" if it fails to find the module it then

searches for it in the whole process to find a comparable process's name to inject it. Once the injection is completed it loads the file and decrypts it to copy the code content into the process and then writes it to the portable executable(PE) file and .exe file that is the MZ file. The .exe file starts the process by praising the PE file and then creates an entry point for the process, after confirmation that there will be no problem to overwrite the entrypoint it searches for the piece of code from the process of Ntoskrnl.exe or Ntkrnlpa.exe.

These codes are written to allocate virtual memory to the given parameters which change the permissions of the entrypoint process from "read only" to "copy on write". It creates a buffer which is of equal size to the Stuxnet file and copies all the related files to this newly created buffer. To complete the process it then writes the place of the newly created buffer in the specific location in the PE module where we have previously copied the data. At the end it rewrites the Modified entrypoint with the code which was originally stored in the memory and to cover up the tracks it calls DeviceIoControl, it sends a request for Io packet to the driver in order to reset all the permissions back to the original state.[4]

IV. CONCLUSION

One who gains access to any system by exploiting any vulnerability has a very clear intention in mind that either they will do any specific task and would then leave the system covering up all the tracks or they will hide in the system in order to attack again. Stuxnet was targeted to setback the Nuclear Programme of Iran.

For Stuxnet most reports say that the attack of the Stuxnet was one of the most sophisticated malware attacks ever recorded till now. Not only because it took the admin rights without coming into the picture but also the facility which it targeted was a highly monitored facility, and it stayed in those systems for significant time. So also the wide area exposure which it took over due to its mechanism was also remarkable.

This also marks the tragic fact that even if the facility had an Air-gapped network still the Stuxnet made such large scale damage to it. Moreover it spread beyond the facility which can be considered as the side effect of such cyber-weapons, like even if they are targeted to a specific purpose there is no way that we can restrict them to the specified area.

In addition to this the Stuxnet also highlighted the weak security standards of the SCADA systems which are used by most of the industries.

As these types of attacks are now gaining popularity in these times the researchers now have more scope to find and stop their spread by developing more rigid systems. And one must make sure that rigid standard operating procedures are followed for those who have access to such crucial systems.

There is future scope for this, as even if the Stuxnet is discovered its spread might not be completely erased. And successors of this malware are supposed to be more sophisticated in terms of camouflaging their presence which will obviously be a great task to deal with.

ACKNOWLEDGEMENT

I would like to extend my warm gratitude to my Department of Computer Engineering and Mrs. Rucha Murkewar (M.E., Assistant Professor, (Structure) Civil Engineering), Smt. Kashibai Navale College of Engineering, Pune-041, for giving me the idea to make this research paper. I also thank my mom and dad who gave me the unparalled support.

REFERENCES

[1]. Crucial Industries ConFront Cyberattacks, McAfee and the Strategic And International studies: In the Dark, 2011.
[2]. David Albright, Paul Brannan, and Christina Walrond (Institute for Science and International Security): Did Stuxnet Take Out 1,000 Centrifuges at the Natanz Enrichment Plant?, December 22, 2010.
[3]. Stuxnet: The computer virus that prevented and started the next world war. : The Why Files (youtube.com), March 4,2021
[4]. Amr Thabet : Stuxnet Malware Analysis Paper
[5]. Risk and Resilience Team Center for Security Studies (CSS), ETH Zürich: Hotspot Analysis: Stuxnet, October 2017.
[6]. Nicolas Falliere, Liam O Murchu,and Eric Chien: W32.Stuxnet Dossier (version 1.3), November 2010.
[7]. Phillip Porras, Hassen Saidi, and Vinod Yegneswaran: An Analysis of Conficker's Logic and Rendezvous Points, February 4, 2009.
[8]. Kaspersky Threats: WORM.WIN32.Stuxnet
[9]. ESET: Stuxnet Under Microscope
[10]. Geoff Chappell: The MRXCLS.SYS Malware Loader, October 21,2010.

TERMS AND THEIR MEANINGS

[11]. Target: Centrifuges used in the uranium enrichment process in a nuclear plant in Natanz in Iran.

[12]. Tool/weapon: Stuxnet: a worm using four zero-day vulnerabilities and infecting computer networks through USB flash drives.
[13]. Air-gapped network: An air-gapped network is isolated from unsecured networks, meaning that it is not directly connected to the internet, nor is it connected to any other system that is connected to the internet. A true air gapped computer is also physically isolated, meaning data can only be passed to it physically (i.e. via USB, removable media or a firewire with another machine).
[14]. FEP: Fuel Enrichment Plant.
[15]. SCADA systems: Supervisory Control And Data Acquisition systems.
[16]. Zero-day exploit/vulnerabilities: Security vulnerabilities of which software developers are not aware and which can be used to hack a system.
[17]. PLC: Programmable Logic Controller.
[18]. DLL: Dynamic Link Library.
[19]. P2P (Peer to Peer): In peer-to-peer (P2P) networking, a group of computers are linked together with equal permissions and responsibilities for processing data.
[20]. PE: Portable Executable.
[21]. RPC: Remote Procedure Call.

Atharva Vilas Gorde. "A Study of Stuxnet Malware: Simplification of Its Working Mechanism." *International Journal of Engineering Research and Applications (IJERA)*, vol.11 (11), 2021, pp 01-08.