RHBVS

ROSE SWE's Heuristic Based Virus Scanner

ROSE SWE, Ralph Roth

Table of Contents

```
  _____   ___  _____   _____
  _____    \/    |   _____    \    \    \    /    /    _____/
   |       _/    ~      \    |    _/\       Y      /\_____   \
   |    |    \    Y     /    |    \ \      /   /           \
   |____|_  /\___|_  /|_____    /   \___/ /_____    /
          \/         \/            \/                          \/  (c)  by ROSE SWE
```

💡   Behavior-based detection (also called "dynamic detection")

RHBVS, short for "ROSE SWE Heuristic-based Virus Scanner," is an anti-virus software that employs heuristics to detect malware. Heuristics are problem-solving strategies that rely on practical experience and general rules to find solutions. In the context of antivirus software, heuristics are utilized to identify malware by analyzing its behavior or characteristics, rather than relying solely on a database of known malware signatures.

The ROSE SWE heuristic-based virus scanner functions by scrutinizing the behavior of a program or file and comparing it against a collection of heuristics specifically designed to identify common patterns of malicious behavior. If the program or file exhibits behavior that aligns with one or more of these heuristics, it is flagged as potentially malicious.

Heuristic-based anti-virus software proves effective in detecting new or unidentified threats that have yet to be included in the database of known malware signatures. However, since heuristics are based on general rules, they can occasionally produce false positives, erroneously flagging benign programs or files as malicious.

To ensure robust protection against malware, it is crucial to utilize a comprehensive antivirus solution that incorporates multiple layers of defense, including signature-based detection, heuristics-based detection, and behavior-based detection. This multifaceted approach enhances the overall security posture and minimizes the risk of both known and emerging threats.

Chapter 1. Introducing RHBVS

RHBVS is a DOS virus scanner for DOS file and hybrid viruses using heuristic scanning technologies and also signature-based detection! This means that RHBVS does not need to be regularly updated like a normal virus scanner. RHBVS also uses an intelligent code analyser. Detection modules for batch viruses, trojans, malware, scripting viruses such as Coral Draw, VBS, HTML, Windows Batch (WBT), JavaScript, SHS (Windows Shell Scrap), Powershell, Bash and IRC (Mirc) script worms are also included!

This is currently/was a unique feature - no other scanner can scan e.g. IRC, HTML or VBS worms with heuristics! RHBVS gives you a detailed virus analysis based on the built-in scan engine.

Chapter 2. Why?

RHBVS was mainly written to be a test platform for the product VirScan Plus by ROSE SWE. All improvements done in VirScan Plus improves RHBVS, RMS (replaces the older FindMirc) and vice versa. For this reason RHBVS is limited in flexibility (e.g. checking boot sectors, Windows system memory or the MBR).

Chapter 3. Requirements

- IBM compatible PC with a 80386 CPU and co-processor!

- 620 KB of free memory (BigMem version) or 500 KB of free memory (Overlay version)

- DOS version 5.0 or higher or Windows 32 bit (RHBVS will not run under Windows 64 bit)

If you are interested in sponsoring the porting of RHBVS to Windows 64-bit or Linux, please reach out to us. While there are currently no plans for such a port due to perceived lack of demand and sponsors, we are open to considering it with sufficient support. In the meantime, we recommend utilizing MPScan (available for DOS32, Win32+64, Linux32+64), a multi-platform malware scanner that incorporates both heuristics and signature-based detection, and has comparable or even superior detection rates.

3.1. SmallMem and BigMem Versions of RHBVS

With the release of RHBVS version 7.18, the size of the executable has reached 500 KB. This has led to reports that the program fails to run on systems with insufficient available DOS memory. As RHBVS continues to be actively maintained and enhanced, the executable is expected to grow even larger in future versions, potentially causing further compatibility issues for systems with limited resources.

To address this challenge, we now offer two versions of RHBVS to accommodate different system configurations:

3.1.1. SmallMem Version (Default)

This version is specifically designed for systems with restricted DOS memory. It employs overlay technology, a method that dynamically swaps portions of code in and out of memory as needed, thereby reducing the memory footprint. However, this approach comes with a trade-off:

- **Performance Impact:** The use of overlays makes the scanning process slower compared to the original version.

- **Performance Optimization:** To mitigate this and improve scanning speed, we recommend providing approximately **512 KB of EMS (Expanded Memory)** which the overlay manager can utilize to optimize performance.

The SmallMem version is now the default option and is well-suited for systems with limited memory resources.

3.1.2. BigMem Version

For users with sufficient free DOS memory, the original version of RHBVS, now called BigMem, is still available in the `BigMem` directory. This version retains the full program in memory without the need for overlays, offering the following benefits:

- **Faster Performance:** Without the overhead of swapping code, the BigMem version delivers faster and more efficient scanning.

- **Recommendation:** If your system has enough free memory to support it, we strongly recommend using this version for the best overall performance.

3.1.3. Summary

To ensure compatibility and optimal performance, users should select the version of RHBVS that best fits their system's memory configuration:

- Use **SmallMem** on systems with limited DOS memory and allocate EMS memory if possible for better performance.

- Opt for **BigMem** if memory availability is not a concern, as it provides faster scanning and a more seamless user experience.

By offering these two versions, we aim to provide flexibility for our users while ensuring that RHBVS remains functional and effective on a wide range of systems.

Chapter 4. Terms

4.1. Heuristic (computer science)

In computer science, a heuristic is a technique designed to solve a problem that ignores whether the solution can be proven to be correct, but which usually produces a good solution or solves a simpler problem that contains or intersects with the solution of the more complex problem.

Heuristics are intended to gain computational performance or conceptual simplicity potentially at the cost of accuracy or precision.

4.2. Computer Virus

In computer security technology, a virus is a self replicating program that spreads by inserting copies of itself into other executable code or documents (for a complete definition: see below). Thus, a computer virus behaves in a way similar to a biological virus, which spreads by inserting itself into living cells. Extending the analogy, the insertion of the virus into a program is termed infection, and the infected file (or executable code that is not part of a file) is called a host. Viruses are one of the several types of malware or malicious software. In common parlance, the term virus is often extended to refer to computer worms and other sorts of malware. This can confuse computer users, since viruses in the narrow sense of the word are less common than they used to be, compared to other forms of malware such as worms. This confusion can have serious consequences, because it may lead to a focus on preventing one genre of malware over another, potentially leaving computers vulnerable to future damage. However, a basic rule is that computer viruses cannot directly damage hardware, only software is damaged directly. The software in the hardware however may be damaged.

While viruses can be intentionally destructive (for example, by destroying data), many other viruses are fairly benign or merely annoying. Some viruses have a delayed payload, which is sometimes called a bomb. For example, a virus might display a message on a specific day or wait until it has infected a certain number of hosts. A time bomb occurs during a particular date or time, and a logic bomb occurs when the user of a computer takes an action that triggers the bomb. However, the predominant negative effect of viruses is their uncontrolled self reproduction, which wastes or overwhelms computer resources.

Today (the trend started round 2005), viruses are somewhat less common due to the popularity of the Internet - instead malware, ransomware and Trojans meanwhile dominate.

Malware, short for malicious software, is an umbrella term used to refer to a variety of forms of hostile or intrusive software, including computer viruses, worms, Trojan horses, ransomware, spyware, adware, scareware, and other malicious programs. It can take the form of executable code, scripts, active content, and other software. Malware is defined by its malicious intent, acting against the requirements of the computer user and so does not include software that causes unintentional harm due to some deficiency (e.g. bugs).

Chapter 5. Options and Switches

Please note that command line options in RHBVS are not case sensitive. You can use either the slash "/" or the hyphen "-" to indicate the start of an option. Additionally, options can be configured using the environment variable RHBVS.

```
set RHBVS=...
```

To disable an option set by setting RHBVS=... you can use the "-" at the end of the option!

when you set

```
set RHBVS=/all
```

than you can disable /all with

```
rhbvs c: -all-
```

5.1. Command Line Options

Run RHBVS.EXE with

```
/?  to see the current supported options.
```

Try also

```
/?? or /UNDOC to see a list of the advance options.
```

You can scan as many drives and directories as you want per run.

```
/vb Code Analyzer (past the switches /ANALYZE or /ANALYSE)
```

With this switch RHBVS gives you a detailed description of all the flags the heuristic scan engines have found.

You can use the option

```
/vbk  then RHBVS waits for a key stroke after every analysis.
```

Use the additional option

```
/log  to save the analysis into a log file.
```

5.2. The Option /virsort

A special note about this option.

> This is one of those "undocumented" switch RHBVS supports. With this switch you can sort in viruses AVP/FProt/VSP/DrSolly etc. misses. With this option RHBVS creates a log file suitable for Virsort or Zoo-Sort (utilities meanwhile deprecated). Take a look at the batch file RZOOSORT.BAT which is included in the package!

For more "undocumented" switches try also: `rhbvs -??`

Chapter 6. User documentation

This text file is written in AsciiDoc and has been converted into nice HTML and PDF files. German-speaking users should download the virus scanner "VirScan Plus" (VSPxxxx.*) and read the German documentation there for further understanding.

Chapter 7. Virus classification

RHBVS classifies the different virus types, their code size and the behavior.

The classification has the following scheme:

```
{Virkit:}[Main Class]{.Length{.Minor Class}{.Germs} (Flags)
```

```
-=[ Virkit ]=----------------------------------------
```

```
Viruses created with a virus kit just like
```

```
+ Biological Warfare (BW)
+ DReg
+ Father_Mac
+ GOTH
+ IVP
+ NRLG, Nuke
+ PS-MPC, MPC, G2
+ TPE, MtE, GCAE, RTFM  etc.
+ VCC
+ VCL
+ VLAD
```

```
-=[ Main Class ]=----------------------------------------
```

```
+ Backdoor- Backdoor (Trojan)
+ Bat     - DOS Batch file virus or Trojan
+ Boot    - Boot virus and EXE header infector
+ CSC     - Coral script virus
+ Companion - small companion viruses
+ Crypt   - encrypted virus
+ Fast    - fast infector, like Dark Avenger
+ File    - appending file infector
+ HLLx    - High level language viruses
            x stands for C=companion, O=overwriting, and P=parasitic
+ IIS     - MS Internet Information Server Worm
+ Joke    - Joke/Fun program. This is not a virus.
+ JS      - Java script virus
+ Mini    - larger overwriting file infector
+ MIRC    - MIRC script worm
+ Multi   - Hybrid (multipartite) files and boot infector
+ Poly    - Polymorphic encrypted virus
+ PIRC    - PIRC script worm
+ SillyR  - trivial memory resident file infector
+ Stealth - virus with stealth capabilities (size or file stealth)
+ TSR     - virus stays resident in memory
+ Tiny    - trivial appending file infector (e.g. Danish)
+ Trivial - overwriting file infector (e.g. Trivial.45)
+ WBT     - Windows Batch virus
+ VBS     - Visual Basic Scripting virus
+ VBS+VBS - multiple VBS infections of one host - yes RHBVS can
            detect multiple infections!
+ Win32,  - Windows platform specific virus or Trojan
+ Win95,98
```

```
+ exact virus name, when using the switch /TROJ
+ exact virus name if found by the polymorph decryption engine
  (Hare, MtE, BW, Grief, TPE, Lucky.Gott etc.)
```

```
-=[ Length ]=---------------------------------------------
```

If possible the virus size. If there is a question mark (e.g.) Virusname.438? the code analyzer assumes this as the virus size!

```
-=[ Germs ]=----------------------------------------------
```

If it is a Generation-1 sample.

```
-=[ Flags ]=----------------------------------------------
```

RHBVS uses the following flags as short cuts:

```
A - Anti debugging or anti heuristic code is used
B - can overwrite the boot sector/MBR (used by the payload or
    by a boot sector infector)
D - found a decryption routine (virus seems to be encrypted)
E - Infects EXE headers like Headerbug or Pure
F - suspicious file access
H - uses hardware related instructions - common for boot viruses
I - uses INT 21h calls in a suspicious way
M - memory resident. Code will remain resident or will control
    some of the DOS functions. Typical for resident file infector
O - opens files for writing code into it
R - suspicious relocation code, typical for file infector
T - checks the date or time (usually used for a payload etc.)
U - Virus tries to stay resident in UMB (upper memory blocks)
W - Windows malware or windows shell code
```

```
! - uses at least FCB and/or directory stealth methods
# - is encrypted or uses code to confuse a code analyzer
```

```
Flags will be "compressed" if more than three flags were found.
RHBVS will show them as "flag: number of occurrence", e.g.: R:4
```

7.1. Some terms

In computer terminology, polymorphic code is code that mutates while keeping the original algorithm intact.

Polymorphic code was invented in 1992 by the Bulgarian cracker Dark Avenger (a pseudonym) as a means of avoiding pattern recognition from anti virus software. This technique is sometimes used by computer viruses, shell code exploits and computer worms to hide their presence. Most anti virus software and intrusion detection systems attempt to locate malicious code by searching through computer files and data packets sent over a computer network. If the security software finds patterns that correspond to known computer viruses or worms, it takes appropriate steps to neutralize the threat. Polymorphic algorithms make it difficult for such software to locate the offending code as it constantly mutates.

Encryption is the most commonly used method to achieve polymorphism in code. However, not the entire code can be encrypted, as it would be completely unusable. A small part of it remains unencrypted and is used for the first start and to decrypt the encrypted software. Anti-virus software targets this small unencrypted part of the code.

Malicious programmers have tried to protect their encrypted code from this strategy by rewriting the unencrypted decryption engine each time the virus or worm is spread. Sophisticated pattern analysis is used by antivirus software to find the underlying patterns in the various mutations of

the decryption engine in the hope of reliably detecting such malware.

Stealth: Some viruses try to fool anti virus software by intercepting its requests to the operating system. A virus can hide itself by ensuring that a request of anti virus software to read an infected file is passed to the virus, instead of to the operating system. The virus can then return an uninfected version of the file to the antivirus software, so that it seems that the file is "clean". Modern anti virus software employs various techniques to counter stealth mechanisms of viruses. The only completely reliable method to avoid stealth is to boot from a medium that is known to be clean.

Chapter 8. False Positives

A false positive, also referred to as a false alarm, occurs when a test erroneously indicates the presence of a signal when there is none. False positives can be encountered in various detection algorithms. For instance, in optical character recognition (OCR), the algorithm may identify an 'a' even when there are only a few dots resembling the letter.

When developing such software, there is always a trade-off between false positives and false negatives, where a true match is not detected. This trade-off involves balancing the risk of Type I errors (false positives that wrongly reject the null hypothesis) against Type II errors (false negatives that fail to reject the null hypothesis when it is false) in statistical hypothesis testing.

Typically, there is a predetermined threshold that determines how closely a match should resemble a given sample before the algorithm reports it as a match. By increasing this threshold, the algorithm becomes more stringent in its detection, requiring a closer similarity for an object to be flagged, thus reducing the occurrence of false positives.

As RHBVS is a rules-based heuristic virus scanner, encountering false positives is a normal part of its operation. If you come across false positives, you can send the executable file for verification and improvement of the scanner. It is important to note that on standard installations, RHBVS should not trigger any false positives.

8.1. Known False Positives

Currently RHBVS detects some hacking tools like unHS etc. But no normal user has this stuff on the disk drive - so no action is taken to fix it. Other known false positives are the memory resident DOS anti-virus programs TBAV and FProt. Both are now obsolete and no longer available.

RHBVS flags them as:

```
"D:\WINDOWS\TBAV_WIN\TBSCANX.EXE  Fast.TSR.File (MBIBBMFR)"
```

This means code to stay resident and to intercept file operation like opening or execution of executable files. When looking at the code analyser of RHBVS we see that TBSCANX stays memory resident (M- flags+TSR),

INT 21h sub functions 3D, 3E & 6C which is typical for a fast infector (Fast) and INT 13h sub function 02 which is typical for boot viruses (B- flags). Due to the fact TBSCANX stays resident it relocates (R-flags) to get its address.

> ℹ️ THAT'S ABSOLUTELY RIGHT - SO RHBVS ONLY REPORTS A PROGRAM LOOKING LIKE A STANDARD FILE VIRUS.... :))

8.2. False positives causes by third party software

Ralf Borgmann reported that the DSAV.VxD intercept the "Live Bait Test" and reports an unknown

virus. This is a bug and false positive of the DSAV.VxD - it can be reproduced only by the first start of RHBVS :-))

>>>>>>> Please send me also viruses RHBVS misses. <<<<<<<<<<

Chapter 9. Error Codes

RHBVS uses the following DOS return codes when terminating. You can use them in batch files or tools like Skull Check etc.

```
Error level |  Meaning
------------+-------------------------------------------------
    0       |  RHBVS completed without any error and without
            |  finding any suspicious program!
    1       |  Misc. errors, like video mode or DOS version!
    2       |  The help screen was invoked.
    3       |  A virus was found in memory (by Quick Memory Scan)
    4       |  One of the signatures files (RHBVS.SIG or
            |  VIRSCAN.TRJ) is damaged or the access is denied!
    5       |  An error occurred creating the log file (/LOG=).
    6       |  Not used
    7       |  Path specified to scan: Access denied
    8       |  Insufficient memory/not enough memory
    9       |  VirScan.IRC|VirScan.VBS is missing or corrupt
   10       |  One or more suspicious files have been found!
  11..18    |  DOS error, please report it to ROSE SWE!
    xx      |  Internal error, please report it to ROSE SWE!
  203       |  Heap Error. Not enough memory available for
            |  RHBVS. Please try to unload some programs
  208       |  Overlay manager not installed
  209       |  Overlay file read error
```

Chapter 10. Technology

RHBVS currently uses over 350 modules to detect the various types of computer viruses. RHBVS can also emulate and follow a polymorphic hidden jump to the virus body, for example used in the Nostradamus.3584 (a.k.a. Grief) viruses. All the software modules have been taken from ROSE SWE's VirScan Plus virus scanner.

RHBVS skips files smaller than 32 bytes. The scanner can even detect and emulate anti-heuristic programmed code! RHBVS has a detection rate of over 98% for trivial and mini viruses and over 80% for boot (image files) and hybrid viruses.

The overall detection is (tested on my virus collection):

```
    Version [Samples]   0.01            0.10            1.00 /TROJ
--[Percent]----------------------------------------------------
ITW-Test set Germany   30.1 [412]     36.9 [412]     39.3 [412]
Classified viruses(1)  N/A            64.1 [6037]    66.7 [6119]
Unclassified viruses   22.4 [1867]    27.5 [1867]    29.5 [2020]
----------------------------------------------------------------
```

```
    Version [Samples]   1.03 (1)        1.05 (1)        1.07 (1)
--[Percent]----------------------------------------------------
ITW                    60.4 [379]     60.4 [379]     66.8 [373]
Classified(1)          77.7 [????]    77.3 [6808]    77.9 [8122]
Unclassified(2)        44.1 [2007]    45.6 [2371]    41.0 [1289]
----------------------------------------------------------------
```

```
    Version [Samples]   2.00 (1)        2.02 (1)        2.03 (1)
--[Percent]----------------------------------------------------
ITW                    75.3 [402]     80.8 [647]     80.9 [649]
Classified(1)          82.1 [8122]    84.0 [9284]    84.4 [9215]
Unclassified(2)        45.3 [1289]    48.8 [1296]    49.6 [1203]
----------------------------------------------------------------
```

```
    Version [Samples]   2.04 (1)        2.05  (1)       2.10 (1)
--[Percent]----------------------------------------------------
ITW(3)                 81.2 [649]     84.8 [649]     76.3 [2503]
Classified(1)          84.6 [9301]    85.9 [9408]    85.1 [9553]
Unclassified(2)        48.2 [1480]    50.1 [2532]    49.4 [1042]
----------------------------------------------------------------
```

```
   Version [Samples]   2.11 (1)      2.20 (1)      2.22 (1)
--[Percent]------------------------------------------------
ITW(3)                 84.1 [649]    76.3 [2503]   85.8 [649]
Classified(1)          84.6 [9301]   85.3 [10181]  79.2 [12409]
Unclassified(2)        42.8 [998]    42.4 [1962]   55.3 [978]
------------------------------------------------------------
```

```
   Version [Samples]   2.30 (4)      2.35
--[Percent]------------------------------------------------
ITW(3)                 86.2 [1718]
Classified(1)          76.4 [18329]
Unclassified(2)        84.5 [1438]   86.8 [795]
MIRC scripts           100.0 [1018]  100.0 [1082]
------------------------------------------------------------
```

```
   Version [Samples]    2.50 (July 1999)       3.01 (Jan 2000)
--[Percent]------------------------------------------------
FProt, unique(1)       75.8 [19236/25393]
Unclassified(2)        72.1 [546/757]          88.2 [1871/2122]
AVP, unique            70.2 [10392/14801]      65.1 [9789/15057]
Scripts (IRC, VBS, JS) 100.0 [1233/1233]
------------------------------------------------------------
```

(1) Detectable by F-Prot (includes more than 700 HLL viruses & Trojans!) All viruses are unique (Virsort)!

(2) These are REAL viruses in my incoming directories, which are not scannable by the newest KAV and F-Prot versions!!!

(3) ITW test set based on Joe Wells ITW lists. Included are all ITW file and boot infector Some viruses used by the VTC ITW test bed has been added to the RHBVS ITW test bed as well as some RIMC viruses.

(4) With switches /TROJ and /HIGH

Main goal is to increase the overall detection rate as well as reduce the false positives.

Chapter 11. Bugs & Limits, Future

- RHBVS runs only on DOS environments. Please provide feedback if we should port RHBVS to Windows and Linux. The trade-off of the port will be around 20-30% less detection (non-portable assembler routines used). Please have a look at MPSCan that is a similar portable heuristic scanner for multiple platforms.

- This program can only handle file names with a maximum of 67+12 chars length (including paths) because the MS-DOS box of NT. If you have longer file names (Win95/98/NT: supports IMHO 252 chars) then you have to map your paths. Detection has been added for LAN-Manager, Netware based networks and Microsoft compatible networks.

- RHBVS is currently not able to scan inside archives (ARJ, ZIP, LHA etc.)

- RHBVS cannot run under some debuggers like Soft Ice due to the HackStop security envelope. ;-)

- RHBVS is limited in scanning MS Office documents, boot viruses as well as Win32 executable (PE/NE).

Testing a virus scanner is not an easy task and should be only done by experts on a large virus collection!

Suggested Options for Testing

- File viruses

```
rhbvs <path> /all /high /log=c:\temp\vtc.log
              /trj is default
```

- Boot viruses (on disks/bootable mediums)

> RHBVS is not designed to scan for (old DOS) boot viruses. Use for that task VirScan Plus or the heuristic boot virus checker ChkPc.

Chapter 12. License

Please note the following: RHBVS is distributed as AnyWare, which implies that the program and documentation are fully copyrighted by the author (ROSE SWE). The program is freely available for use in non-commercial environments, similar to freeware. If you find RHBVS beneficial and would like to contribute to its improvement, please feel free to send anything, for example helpful suggestions, such as emails, bug reports, or even monetary support to the designated contact. Your support is greatly appreciated!

Chapter 13. History

13.1. Version 8

14.02.2025 8.44   New AVR modules added and also hundreds of new viruses
                     added

11.01.2025 8.30   Adjusted number of found viruses/malware. New viruses
                     added

13.12.2024 8.00   First version to be public released with reduced memory
                     footprint. Added a lot of new viruses.

13.2. Version 7

10.12.2024 7.63   Due to memory pressure the option /LIST was removed

08.12.2024 7.53   First version that uses Overlay to reduce the memory
                     footprint RHBVS meanwhile has (500 KB EXE file). Added
                     new viruses (AVR, Signatures...).
                     Rewrote this documentation.
24.09.2024 7.00   Added a new AVR module. New viruses added.

13.3. Version 6

04.09.2024 6.93   Consolidated and updated virus signatures. More
                     viruses added. Improved AVR modules.

18.07.2024 6.83   Better IRC detection. More viruses added.

03.07.2024 6.79   Better entrypoint handling, better handling of false
                     positives. New viruses added. Updated documentation.

07.06.2024 6.70   Maintenance update. Small enhancements,
                     new viruses added.

12.03.2024 6.58	Maintenance update	

04.02.2024 6.47	Improved entry point engine. Maintenance update. Small enhancements, new viruses added.	

27.12.2023 6.39	Better detection. New viruses added.	

30.11.2023 6.32	Better VCL detection. Small enhancements, new viruses added.	

10.11.2023 6.26	Small enhancements, new viruses added.	

13.10.2023 6.19	AVR_Mini/Trojan. Maintenance update. Small enhancements, new viruses added.	

25.09.2023 6.18	Small enhancements, new viruses added. Maintenance update.	

28.08.2023 6.17	Maintenance update adding new virus detection	

13.07.2023 6.16	Small enhancements around the AVR-Trivial engine. New viruses added.	

20.06.2023 6.15	Maintenance update adding new virus detection	

05.06.2023 6.14	Maintenance update adding new virus detection	

06.05.2023 6.13	Maintenance update adding new virus detection	

08.04.2023 6.12	Maintenance update adding new virus detection	

23.02.2023 6.11	Maintenance update adding new virus detection	

| 07.02.2023 | 6.10 | Added new *NIX script and batch detection engine to RHBVS with more than 1000 viruses. |

| 31.01.2023 | 6.00 | Rewritten IRC and Batchvirus engine. New viruses. Improved entry point engine. |

13.4. Version 5

| 13.01.2023 | 5.71 | Complete rewritten command line engine. New viruses added. Improved entry point engine. |

| 15.12.2022 | 5.68 | Enhanced AVR:Mini, AVR:Trivial & AVR:Silly engines. Enhanced VirusKit detection. New viruses added. |

| 25.11.2022 | 5.67 | Enhanced AVR modules, new viruses added. |

| 09.11.2022 | 5.66 | No user visible changes. Enhanced code emulation and entry point detection. New viruses added. |

| 01.10.2022 | 5.65 | Added 4 new AVR modules. New viruses added. |

| 11.09.2022 | 5.64 | Small improvements and new viruses added |

| 02.09.2022 | 5.63 | Version bump after adding a lot of new virus signatures. |

| 25.05.2022 | 5.62 | Fixes with UTF-8 formatting. Thus maybe 3rd party scripts may be adjusted filtering the output of RHBVS log file. New viruses added. |

| 15.05.2022 | 5.61 | Small improvements. New viruses added. |

18.03.2022	5.60	Changes and enhancements to the internal scan engine. Added new viruses.
10.02.2022	5.54	Internal changes and improvements. Added new heuristic signatures and malware detection.
23.01.2022	5.52	Added a new AVR engine for DOS Debug script viruses. Added new viruses.
07.12.2021	5.51	New viruses added
09.07.2021	5.50	Internal changes and optimization for the virus databases. Better and faster detection. New viruses added.
24.06.2021	5.40	Internal enhancements. Added new viruses. RHBVS now checks for import files using the file rhbvs.cfg. All files from rhbvs.cfg must be provided.
08.06.2021	5.32	New viruses added. Changed internal database format (virscan.*)
25.05.2021	5.31	New viruses mainly from MPScan added. Small enhancements.
06.03.2021	5.30	Added the detection engine from MPScan to RHBVS.
22.02.2021	5.22	More viruses added.
12.02.2021	5.21	Fixed a few false positives. Added new viruses.
14.01.2021	5.20	Rewritten "MalwareScriptViruses" engine, therefore all virus databases require the internal version 4.00 or higher. New viruses added.

29.10.2020	5.15	Added more than 700 viruses. Updated the AVR Modules again. Added more signatures for boot and multipartite viruses.

16.10.2020	5.14	Added around 300 viruses. Changes around the AVR modules (Tiny, Trivial, Mini). Enhanced documentation.

13.10.2020	5.13	Added around 400 viruses. Changes around the AVR modules.

05.10.2020	5.12	Massive changes and enhancements around the AVR modules. Added hundreds of viruses.

19.09.2020	5.11	New viruses. Added 4 new heuristic search modules

01.09.2020	5.10	Added new viruses.

01.07.2020	5.09	Added new viruses.

27.03.2020	5.08	Added new viruses.

08.12.2019	5.07	Added new viruses. More heuristic detection.

01.11.2019	5.06	Some internal changes. Added new viruses.

22.06.2019	5.05	Added new viruses. Documentation update.

29.12.2018	5.04	Added new viruses. Documentation update.

20.09.2018	5.03	Added 22.000 viruses.

28.03.2018 5.02	This documentation was ported to AsciiDoc.

06.12.2017 5.01	Small enhancements. Major reprogramming of the signature based detection.

29.11.2017 5.00	Trojan detection is not compatible with pre 5.00 releases. New viruses detection added.

13.5. Version 4

27.11.2017 4.98	Public release with new viruses detection.

09.09.2017 4.97	Public release. Enhancements and new viruses.

15.02.2017 4.96	Enhancements and new viruses.

22.04.2016 4.93	Public release. Enhancements and new viruses.

20.04.2015 4.92	Public release. Enhancements and new viruses.

10.11.2014 4.91	Public release. Enhancements and new viruses.

30.12.2013 4.90	Generic encrypted script detection added. Enhancements and new viruses.

30.10.2013 4.84	Enhancements for better detecting Win32 and Win64 viruses. Added new viruses.

03.03.2013 4.83	5000 viruses added, changed home page URL

30.09.2012 4.81	Small enhancements, new viruses.

16.10.2011	4.80	New viruses added, esp. the German "Staatstrojaner" (file+live test).
07.06.2011	4.79	New viruses added. Enhancements for Win32, Dos32 and Linux console output.
03.02.2011	4.78	New virus detection added. Fixed an run-time error bug.
13.08.2010	4.77	Added a lot of windows malware and windows shellcode detection stuff. New viruses added.
18.06.2010	4.76	Win32.Shellcode handler improved. VBS encrypted detection improved.
13.04.2010	4.75	New viruses added. New icon for RHBVS. Documentation updated.
19.03.2010		Major update/enhancements added to PeHead.
14.03.2010	4.73/4.74	Small enhancements and new viruses added.
19.02.2010	4.70-4.72	Small bug fixes and enhancements. New viruses added.
30.03.2009	4.68/4.69	Massive enhancements around the /rename function. Bug fixes and new viruses added.
06.02.2009	4.67	Small enhancements for Windows Vista. New viruses added.
16.11.2008	4.66	Small bug fixes and enhancements. New viruses added.

11.01.2007 4.65   Changes on the /Rename functions.

30.09.2006 4.64   Enhancements, new viruses. Changed
                        virus database.

09.08.2006 4.63   Small enhancements (e.g. .PNG detection).

25.04.2005 4.62   Enhanced the docs. Added new signatures
                        to the heuristic scan engines.

10.03.2005 4.60   Changed and enhanced the internal database.
                        Added new scan engines and viruses.

06.01.2005 4.51   Enhanced VBS engine. New viruses added.

13.11.2004 4.50   Added new viruses.

19.08.2004 4.50-RC2   Added ~600 new viruses. Fixed a few
                        false positives.

17.08.2004 4.50-RC1   Complete redesign of the script scanning
                        engines (VBS, Script, IRC, Batch etc.).
                        A lot of new viruses added.
                        The signature files (virscan.*) are not
                        compatible with the 4.1x and below
                        releases!

16.06.2004 4.13   Small fixes, 400 viruses added.

14.04.2004 4.12   Added QWTC - "Quick Windows Trojan Check"

21.01.2004 4.11   Bug fixing of the command line handling
                        engine. New viruses added.

09.09.2003 4.10   Bug fixing, RHBVS now requires a
                        co-processor.

07.09.2003 4.05   Added and enhanced some scan engines
                        and added tons of new viruses. Bug fixes.
                        (EXE file is therefore 20 KB bigger!).

06.09.2003 4.02   Ported and enhanced some of the scan
                        engines to Linux. New viruses added.

16.07.2003 4.00   New viruses. Changed the internal Trojan
                        and malware engine to run on Linux too.

13.6. Version 3

13.05.2003 3.96   Added tons of new viruses.

25.03.2003 3.95   New and enhanced engines for VBS viruses.

27.02.2003 3.94   Fixes for HMA/A20 gate check. Added tons
                        of new viruses.

07.11.2002 3.93   Added tons of new viruses.

05.11.2002 3.92   Added new viruses, therefore internal hash
                        tables had to be adjusted.

03.11.2002 3.91   Build 433
20.08.2002 3.91   Build 423
18.06.2002 3.91   Documented the switch /OnlyFull. Added
                        new viruses.

05.05.2002 3.90   New viruses added. Changed the format of
                        Virscan.trj

```
25.04.2002  3.81        Added new viruses. Fixed a false positive.
23.04.2002  3.80        Fixed a bug with Win2000/NT. Changed the
                        signature files.
```

```
19.04.2002  3.73        Added 120 viruses.
11.04.2002  3.72        Added 300 viruses.
```

```
17.03.2002  3.71        Changed documentation (also renamed from
                        *.DOC to *.TXT). Added new viruses.
```

```
22.01.2002  3.70        New viruses. DOCS changed. Bundled with Win32
                        installer.
```

```
09.01.2002  3.64        New viruses. Added .PIF file for Win9x.
```

```
10.12.2001  3.63        New viruses added. New option -delYN added.
```

```
08.10.2001  3.62        New viruses added.
```

```
15.08.2001  3.61        New viruses. New generic scan engine for
                        IIS-Worms added. Should find every worm
                        that uses the IIS Backdoor. To scan for
                        such worms, you currently need the option -ALL
```

```
30.07.2001  3.60        Added 300 new batch and script viruses
                        using the new designed scan engines from
                        RHBVS 3.55. Those signatures are stored
                        in the new file "VIRSCAN.IRC".
```

```
26.07.2001  3.55        Added tons of new viruses. Added .LNK as
                        default extension. Introduced a version
                        numbering to VIRSCAN.VBS (needed for new
                        generic script detection). Added generic
                        script detection engine. Added new engines.
```

08.06.2001	3.51	New viruses added. Better detection of anti heuristic programmed VBS viruses.
03.05.2001	3.50	Depending on your machine (386, 486 etc.) and operating system, RHBVS is now up to 20 percent faster. New viruses.
16.03.2001	3.45	Added more than 100 new VBS viruses. Added .JSE, .VBE, .WSH as a default extension. Included on the fly decryption of MS VBS encrypted files (.VBE). New signatures added. VBS scan engine updated.
17.02.2001	3.41	Update of the VBS scan engine to find VBS.NeueTarife/AnnaKov. New viruses.
19.01.2001	3.40	New viruses added (of course :). Option -ShowErr added. Statistic enhanced (+ time, + total errors). Some false positives fixed. We have ported parts of the scan engines to win32. As a benefit the scanning is now much faster due to the enhancements we had to do for the porting.
04.01.2001	3.32	Added four new scan engines, VBS engine was enhanced. 70 new viruses added.
27.11.2000	3.31	New viruses added.
14.09.2000	3.30	Added Win32 Stealth Bait test. New viruses added.
25.07.2000	3.21	Added .VBA as default extension. /RenPE enhanced. New viruses added.

05.07.2000	3.20	Faster scanning due to rewrite of the VBS and MIRC analyzer Add option /NoScript (same as /NoVBS). New viruses. Added 180 Trojans. Added MS Mail scanning (MSFT). Added generic VBS detection (construction kits etc.). Added generic Batch file detection.
22.06.2000	3.11	Added detection for 680 Backdoors. 20 new VBS viruses added. Added .VXD and .SHS as default extensions. Added 70 Trojans. SHS will now be scanned too (VBS.Life_Stages).
26.05.2000	3.10	Added detection for 250 Win/Win32 Trojans, Backdoor and password stealing programs. Added detection for 20 new VBS viruses. Added .DLL extension as default. New viruses.
07.05.2000	3.03	Due to the various VBS.Love-Letter variants we added to the virus name additionally the length. When you use RZOOSORT.BAT to sort your Love-Letter variants, they go now in separate directories.
28.04.2000	3.02	Added MIRC detection in .PIF files. Added option /NoVBS. /NoVBS is also set if VIRSCAN.VBS was not found! New viruses :) Added options /NoTrj and /NoTroj
29.01.2000	3.01	Added HLP, AVI, CHM, FTS, CNT detection. Added Joke class to RHBVS. New viruses :) Changed the VBS detection engine for the first anti RHBVS specific viruses.
03.12.1999	3.00	Added ACE and (WAV) Wave detection. Added "T" flag (time/date). Added 750 new viruses. Added new scan engines. Added the options /VB, /VBK (code analyzer) and /REPORT. Better Java script detection added. Nicer screen output. The switch /stdout is now obsolete and not supported any longer!

13.7. Version 2

```
01.09.99  2.56          Added ARJ and LZH archive detection.
                        Renamed /ANALYSE to /WHOLE (planed to
                        add switch /ANALYSE[=language.dat]).
                        RHBVS can now handle multiple infections of
                        VBS viruses.
```

```
11.08.99  2.54          New viruses. Tested RHBVS under Win2000b3
                        Server and fixed all bugs.
```

```
24.07.99  2.52          Added new VBS, JS and MIRC viruses using a
                        new detection engine.
```

```
18.07.99  2.51          WBT (Windows Batch) virus class added.
                        New viruses added.
```

```
10.07.99  2.50          HTML, JS, CS and VBS detection added. New
                        viruses and other malware added.
```

```
24.05.99  2.35          Approx. 500 viruses added. Basic PIRC, INF
                        and VBS detection added. Option /COMP
                        (generic companion detection) added.
```

```
17.02.99  2.34          Option /NOMEM added. New viruses.
                        Added detection of HTML, PDF (Adobe Acrobat)
                        and MDB (MS Access) file format.
```

```
15.01.99  2.33          Option /RAW added. Bug with long directories
                        under Win-NT fixed. Tons of new viruses and
                        Trojans added. Added Natas decryption engine
                        from VSP. Enhanced the rhbvscum.awk script.
```

```
02.01.99  2.32          Command line handling improved. Mirc detection
                        improved. Code analyser and option /Virsort
                        enhanced. New viruses and Trojans added.
                        File sharing handling for Windows enhanced.
```

29.12.98	2.31	Fixed some bugs and false positives. Enhanced the Mirc classification. Added the rhbvscum.awk script to the package.
29.11.98	2.30	Added Mirc script worm detection and heuristics. Improved file handling. Improved /RENAME capabilities. New viruses and Trojans If VIRSCAN.TRJ is found automatically option /TROJ is added!
20.10.98	2.24/2.25	Non public releases!
24.08.98	2.23	New viruses and Trojans Added a new Trojan detection. Added new entry point detection. Bug fixes. RHBVS uses now the same "smart renaming" engine like RFW. SYS virus detection added.
17.05.98	2.22	New viruses. Added new scan engines (VCL, Mini, Trivial etc.).
07.04.98	2.21	Fixed a lot of minor bugs in the /Rename section. Better Live Bait Test. RZOOSort changed. Added a new internal scan engine. Tons of new viruses added :)
18.03.98	2.20	/Rename, /Renumber now support more Excel formats (.XLA, .XLS etc.), credits: A. Marx Added advanced check for resident stealth viruses (Stealth Live Bait Test). Added more than 40 boot viruses and more than 70 file infector Improved the boot heuristics. Minor bug fixes. Currently I am working on a neural network for RHBVS so it many take a time for the next release :-))

15.02.98	2.11	Added or fixed the following features:
		+ Added more than 50 new viruses.
		+ Fixed some false positives (R. Borgmann)
		+ More compatible file access. Credits
		  (Christian Ghisler & Ralf Borgmann).
		+ Added new search engines and flags.
		+ RHBVS can now only be aborted with the
		  Escape key (SR by R. Borgmann).
		+ Heuristic flag compression/sorting
		+ /Renumber=Value switch now works
		  correctly (one of those undocumented
		  features :-))

29.01.98	2.10	Enhanced check for stealth viruses and fast infector added. Added 350 new viruses. Enhanced companion detection. Enhanced boot virus detection. Added new search engines. Improved the statistics. Enhanced code analyser Fixed some false positives. Added the batch file RZOOSORT.BAT to the package. RHBVS does now a much better classification of the virus using his new code analyser. Changed the heuristic to produce less false positives than the 2.05 release.

28.12.97	2.04	Now the /LOG switch supports file names, e.g. /LOG=C:\TMP\RHBVS.NEW etc. Changed the error level (DOS return codes) and documented them in RHBVS.DOC. New viruses added, fixed some false positives and bugs. New flag "A" added. Added the new virus group "Poly". Added an entry point resolver for the _310 virus. AVR for boot viruses enhanced and improved. Sanity (integrity) self check added!

13.12.97	2.03	Fixed again some false positives received from Ralf Borgmann. About 230 new viruses added. Now the signatures file RHBVS.SIG also contains flags. Added new search engines. Modified the live bait test to fool the DSAV.VxD.

21.11.97	2.02	Fixed about 10 false positives (credits Ralf Borgmann). Added new search engines and new viruses. Overall detection ratio is now 84 percent!
08.11.97	2.01	Fixed two false positives. Added more than 20 new scan engines. Enhanced the Mini and Trivial scan engine. Added more than 200 viruses! RHBVS now scans also files with the extensions .IMG, .BOT and .BIN.
01.11.97	2.00	Added the option /LOG to generate a simple log file. Added more than 80 new scan engines - they are the compressed and optimized search strings from VirScan Plus.

13.8. Version 1

12.10.97	1.07	Added new viruses. Added a new entry point detection for the _1015 virus.
20.09.97	1.06	Windows NT compatibility enhanced. Added new viruses.
09.08.97	1.05	Added some viruses and a new entry point detection engine for the Demo Fraud virus. Windows-NT compatibility enhanced. Added a PIF file for Windows NT 4.0.
13.07.97	1.04	Added the switch /FILETYPE. Added a check for corrupted files. Added a few new viruses. Fixed some false positives.
06.07.97	1.03	Enhanced the Mini-AVR module. Added new viruses. Fixed some minor bugs. Added option /HEUR. Release for SAC ftp etc.

```
28.06.97  1.02          Fixed two false positives. Added a few
                        viruses. Changed the help screen.
                        Added one search engine for EXE-Header
                        viruses. Changed access mode for faster
                        accessing write protected discs. Added
                        the 'E'-Flag.
```

```
11.06.97  1.01virnet    Changed some DOCS. Release for Virnet.
09.06.97  1.01          Added the Option /CONT and /HIGH.
                        Enhanced one search engine to find the
                        Make2 virus.
```

```
07.06.97  1.00          Added the option /TROJ.
                        Improved the Tiny code analyser, added the
                        flags 'H' and '#'.
                        First official release
```

13.9. Beta Versions

```
29.05.97  0.10          Improved the detection rate more than 5%!
```

```
27.05.97  0.02          Added the option /AUTO and /BEEP.
                        Added RHBVSGER.FAQ, enhanced the DOC.
                        Fixed a bug when redirecting the output
                        using the stdout option (rhbvs -stdout>file)
                        Detection on exe packers added.
```

```
22.05.97  0.01          Initial release
```

Chapter 14. Credits

People who helped to improve this product or have given feedback.

🛈   In alphabetical order

```
Andreas Haak            code analyser & more
Andreas Marx            technical consultant :)
Axel Pettinger          Mirc stuff
Bert De Rijck           Fam_????
Carsten Kruse           Mr. "enhancements"
Christian Ghisler       technical consultant :)
Claus Vogt
Frank Ziemann           Backdoor, Trojan and Worm testing
Hanno Boeck             Mr. "false positives" :)
Jerry Hodges            CRC32
Joe Hartmann            Mirc, false positives, RIMC project
Joerg Abdinghoff        initial idea for /ANALYZE, now /vb or /vbk
Lukas-Fabian Moser
Laurent Gerard          new virus
Mano Schwarz
Mathias Brunner
Masterball/codeBreaker  HMA/A20 testing
Michael Hering          checksum, FP, RHBVS.DOC, easily switches
Nobert Kirch            stdout bug
Peter Kosinar           FP, missed viruses
Ralf Borgmann           Mr. RHBVS beta tester :)
Robert Flogaus-Faust
Sebastian Boehm
Stonehead               Mr. "false positives" :)
Tjark Auerbach          DOX
Toralv Dirro            RIMC project
Valentino Tosatti       Mr. "false positives" :)
Veit Kannegieser
```

You? ..

Chapter 15. Files

```
CRCHECK.TXT    checksum file of the whole distribution
ROSEBBS.TXT    the author's address and ROSE support BBS, WWW etc.
```

```
FILE_ID.DIZ    short description of the package
  RHBVS.XXX    checksum file for integrity check
  RHBVS.MSG    Message/language file for switch /vb
  RHBVS.DOC    this documentation
  RHBVS.EXE    the main executable
  RHBVS.PIF    Win 3.1/9x/NT/2000 program interface file :-))
  RHBVS.OVR    Overlay file for RHBVS (SmallMem version)
```

```
  RHBVS.SIG    some heuristic scan engines and flags
VIRSCAN.TRJ    signature file for HLL viruses and Trojans
VIRSCAN.IRC    signature file for script and batch viruses (IRC, BAT...)
VIRSCAN.WSM    signature file for script viruses (IRC, VBS, JS, CSC...)
               [windows scripting malware]
VIRSCAN.MPV    signature file for multipartite DOS file viruses and
               boot/MBR viruses. Contains also signatures for file viruses
```

```
RHBVSCUM.AWK    AWK script to create statistics reports from RHBVS.LOG
RZOOSORT.BAT    handy batch file to sort your unknown viruses!
```

Chapter 16. Miscellaneous

> 💡 Why is RHBVS.EXE such a small program? Well it is compressed using a so called online compressor. Here are the results finding the best compressor for RHBVS.EXE

```
Original size (10.06.2000) =    342.560 bytes    **
           (17.07.2003) =    385.120 bytes
           (09.09.2003) =    396.928 bytes
           (25.04.2005) =    407.344 bytes
           (10.02.2007) =    410.944 bytes
           (13.08.2010) =    413.280 bytes
           (28.11.2017) =    415.664 bytes (147kb compressed)
           (28.03.2018) =    416.832 bytes (132kb compressed)
           (29.10.2020) =    432.640 bytes (->  136.724)
           (02.09.2022) =    462.752 bytes (161kb)
           (20.06.2023) =    486.032 bytes (167kb)
           (08.12.2024) =    503.392 bytes (180kb)
```

When RHBVS reached 500 KB (Version 7.18) we decided to reduce the memory footprint of RHBVS by using Overlay technology.

Computer Viruses and Malware - A Short Overview

A computer virus is a piece of code (software) that is installed on a computer either by a hacker, by another compromised computer (replication), malicious attachments/mails or a website (drive-by infection). It performs functions that the computer owner does not authorize and does not want.

Viruses are sometimes also referred to as malware. This is usually where they have adverse effects on the computer user, such as logging each keystroke (through a keylogger), audio recording or snapshots of each screen.

Such infection can lead to identity theft, endangerment of bank or purchase card data or loss of confidential data. It is more likely to occur on home computers that are normally not as security managed as corporate computers.

Chapter 17. Malware

Malware, or malicious software, is a generic term for a variety of malicious or intrusive software, including computer viruses, worms, Trojans, ransomware (ransoms), spyware, adware, scareware and other malicious programs. It can take the form of executable code, scripts, active content and other software. Malware is defined by its malicious intent, which violates the requirements of the computer user - and therefore does not include software that causes unintentional damage due to a defect.

Programs officially delivered by companies can be considered malware if they secretly violate the interests of the computer user.

Chapter 18. (Computer) Virus

A computer virus is a type of malicious software program ("malware") that, when executed, replicates itself by modifying other computer programs and appending or inserting its own code. When this replication succeeds, the affected programs are then said to be "infected" with a computer virus.

The term "virus" is also commonly, but erroneously, used to refer to other types of malware. "Malware" encompasses computer viruses along with many other forms of malicious software, such as computer "worms", ransomware, spyware, adware, Trojan horses, keyloggers, rootkits, bootkits, malicious Browser Helper Object (BHOs) and other malicious software. The majority of active malware threats are actually Trojan horse programs or computer worms rather than classic computer viruses.

Roughly you can distinguished between - Memory resident (fast) infecting viruses and - Direct action viruses

18.1. Direct Action Viruses

Direct action viruses are a type of malware that infect individual files on a computer, rather than the boot sector or Master Boot Record (MBR). They are called "direct action" viruses because they are executed each time a specific file is opened or executed, which allows the virus to infect other files on the computer.

Some of the simpler computer viruses do not actively manifest themselves in computer memory. The very first file infector viruses on the IBM PC, such as Virdem and Vienna, belong to this category. As a rule, direct viruses do not spread quickly and are not easily spread in the wild.

Direct action viruses load themselves into computer memory with the host program. Once they have taken control, they search for new objects to infect by searching for new files. For this very reason, one of the most common types of computer viruses is the direct action infector. This type of virus can be created relatively easily by the attacker in binary or scripting languages on a variety of platforms.

Direct action viruses typically use a FindFirst, FindNext sequence to search for a number of victim applications to attack. Typically, such viruses only infect a few files when executed, but some viruses infect everything at once, enumerating all the directories for victims.

Direct action viruses typically spread by attaching themselves to executable files, such as .exe, .com, or .bat files. When an infected file is executed, the virus infects other files on the computer and may also cause other malicious activity.

18.2. (Computer) Boot Virus

Boot viruses are the oldest known computer viruses. They were the most common type of virus until 1995, but are now extinct. Today, there are almost no boot sector viruses anymore because BIOS and operating systems usually have well-functioning software or hardware protection.

A boot virus is a computer virus that becomes active when the computer starts (boots) before the operating system (DOS, Linux or Windows) is fully loaded. Boot sector viruses take advantage of the fact that the boot sector is always loaded first. On floppy disks, the virus is at least partially in the boot sector, so even floppy disks with no files on them can be infected. On hard disks, the virus infects the master boot record (MBR) or logical boot sector.

A boot sector virus infects the boot sector of floppy disks and the master boot record (MBR) of a hard drive. The boot sector is the first physical part of a floppy disk and is a sector (512 bytes). The boot sector is used by boot floppies to boot from the floppy. When a user tries to boot from an infected boot floppy, or leaves an infected floppy in the floppy drive when the computer starts up, the BIOS accesses this sector and executes it with the appropriate BIOS boot setting. The virus then attempts to infect the hard disk's MBR every time the computer is started. When an infected computer is started, the MBR, which is normally responsible for recognising the different partitions on the hard drive, is loaded. Once loaded, the virus remains in memory and monitors access to floppy disks. When a floppy disc is inserted into a computer infected with a boot sector virus, the virus infects the boot sector of the floppy disc.

Known boot viruses include the Form virus, Parity Boot and Boot-437.

18.3. Multipartite Virus

A multipartite virus is a computer virus that infects and spreads in multiple ways. The term was introduced to describe the first viruses that included DOS executable files and PC BIOS boot sector virus code, where both parts are viral themselves. Prior to the discovery of the first of these, viruses were categorized as either file infectors or boot infectors. Because of the multiple vectors for the spread of infection, these viruses could spread faster than a boot or file infector alone.

The first virus that infected COM files and boot sectors, Ghostball (more a dropper than a real multipartite virus), was discovered by Fridrik Skulason in October 1989. Another early example of a multi-part virus was Flip, Frodo, Delwin and Tequila. Tequila for example could infect both DOS EXE files and the MBR (master boot sector) of hard disks.

Chapter 19. Trojan horses

A Trojan horse is a program that does something undocumented which the programmer intended, but that users would not accept if they knew about it. By some definitions, a virus is a particular case of a Trojan horse, namely, one which is able to spread to other programs (i.e., it turns them into Trojans too). According to others, a virus that does not do any deliberate damage (other than merely replicating) is not a Trojan. Finally, despite the definitions, many people use the term "Trojan" to refer only to a non-replicating malicious program.

Chapter 20. Ramsomware

Ransomware is a particularly invasive form of malware that hijacks a victim's data or device and holds it hostage (or makes false claims of illegal activity, pornography use, or suggests a system is already infected with viruses) until a sum of money is paid to secure its release.

20.1. Introduction to Ransomware

Ransomware is a type of malware that hijacks a victim's data or device, demanding a ransom—usually in cryptocurrency—for its release. Sometimes, it also makes false claims about illegal activity or pre-existing infections to pressure victims. Paying the ransom doesn't guarantee recovery, as attackers may refuse to decrypt the data or demand more payments. According to Statista, only 54 per cent of organisations regained access to their data or systems after the first payment in 2021. Paying the ransom also encourages attackers to continue their malicious activities. In addition, the vulnerability still exists and can be exploited by another criminal group.

Ransomware has become a global threat, with attacks growing more sophisticated. Understanding its history, evolution, and tactics is critical to combating it.

20.2. Origins of Ransomware

Ransomware first appeared in 1989 with the DOS-AIDS Trojan (PC Cyborg), created by Joseph L. Popp. Distributed via floppy disks labeled "AIDS Information," it used simple encryption and demanded $189 for unlocking files.

Popp exploited public fear of the AIDS epidemic to spread the malware. Despite its basic design, many fell victim, incurring financial losses and data breaches. Popp's erratic behavior led to his arrest, but he was deemed mentally unfit to stand trial. This event set the stage for ransomware to evolve into a significant cybercrime.

20.3. Evolution of Ransomware

Ransomware has transformed into a sophisticated, profitable industry:

- **Targeted Attacks:** Criminals now target high-value victims like corporations, hospitals, and government agencies.
- **Stronger Encryption:** Modern ransomware uses advanced encryption, making decryption without a key nearly impossible.
- **Cryptocurrencies:** Bitcoin and Monero enable anonymous payments, fueling ransomware's growth.
- **New Tactics:** Techniques like double extortion (encrypting data and threatening leaks) and Ransomware-as-a-Service (RaaS) allow even non-experts to launch attacks.

Notable attacks like WannaCry and NotPetya caused massive global disruptions, targeting businesses and government agencies.

20.4. Modern Ransomware Characteristics

Ransomware attacks today are more targeted and damaging:

- **Anonymous Payments:** Cryptocurrencies make tracking transactions difficult.

- **Time Pressure:** Deadlines threaten data deletion or leaks, forcing quick victim responses.

- **Unreliable Decryption:** Many victims don't recover access, even after paying. In 2021, only 54% regained access after the first payment.

- **Encouraging Crime:** Paying ransoms perpetuates these attacks and leaves vulnerabilities open for future exploitation.

20.5. Steps in a Ransomware Attack

Ransomware attacks follow a common progression:

1. **Gaining Access:** Attackers infiltrate systems using phishing emails, malware downloads, or exploiting vulnerabilities.

2. **Spread:** Malware spreads within the network, either automatically or manually in targeted attacks.

3. **Hostage Taking:** Data is encrypted, and a ransom note demands payment for decryption or preventing data leaks.

20.6. Mitigation and Defense

Defending against ransomware requires proactive measures:

- Stay vigilant against phishing attempts and suspicious files.

- Regularly update systems and software to patch vulnerabilities.

- Use multi-factor authentication to secure accounts.

- Maintain offline backups of critical data.

- Invest in robust cybersecurity tools and training.

20.7. Ramsonware: Conclusion & Best Practices

Ransomware is an ever-evolving threat. Staying informed and implementing strong defenses can help mitigate the risk and minimize impact.

Ransomware attacks can be extremely damaging and complex, and the timeframe for action is very limited. The best way to deal with them is to avoid them in the first place, and use mechanisms to prevent and mitigate their impact. The best way to prevent a malware attack is to follow good operational and security practices, such as

- Keep all software and operating systems up to date.

- Use anti-virus and anti-malware software on desktop systems.

- Regularly scan for vulnerabilities and comply with security policies, the key is to do this regularly.

- The best way to do this is to automate it so that it does not become a problem and can be integrated as part of the deployment process.

- Ensure that the software supply chain is properly secured. From an attacker's perspective, attacking the supply chain may be the easiest way to reach most, if not all, of an organisation's systems.

- Implement proactive measures and adopt a zero-trust policy. This applies to containers as well as traditional environments.

- Implement password validation best practices, such as avoiding common words and using long phrases that are easier for humans to remember but harder for machines to crack.

- Educate staff on basic security principles, such as being wary of suspicious emails, recognising suspicious links and managing data to avoid storing critical data in unsecured locations.

- Perform regular backups and always keep a cold backup in a separate physical location with no network access. Ensure that recovery procedures are tested regularly.

- Automate the provisioning of your infrastructure so that you can restore your systems quickly - time is money.

- Have a disaster recovery plan in place and ensure it is tested regularly.

Chapter 21. Malicious Mining Software (Crypto-Miner)

Starting in 2018 Malware authors are increasingly relying on malicious mining software. This year for the first time there have been more infections of this type than with ransomware. More and more online criminals seem to turn their backs on ransomware and rely on crypto-miner. They secretly dig crypto money on infected computers - Monero is particularly popular. This is obviously extremely lucrative, as the latest figures show.

Reasons for the turnaround? If a ransomware/Trojan strikes and encrypts data from victims, they usually have to pay a ransom in the form of bitcoins. This is an obstacle that not every victim can or will take. Crypto-miner, on the other hand, only needs to infect computers. Afterwards, they dig in secret without any sacrifices and make silently sure that they bring the authors big profits - and not too short when you look at the exploding prices of different crypto currencies.

Nowadays even commercial antivirus software tries to use the user computer when idle for mining. So this kind of software is both a malware scanner and malware itself :-(

Chapter 22. Greyware

Grayware (or greyware) is a general term sometimes used as a classification for applications that behave in a way that is annoying or unwanted, but less serious or problematic than malware. Grayware includes spyware, adware, dialers, joke programs, remote access tools and any other unwanted files and programs other than viruses that are designed to affect the performance of computers. The term has been in use since at least September 2004.

Grayware refers to applications or files that are not classified as viruses or Trojans, but can still affect the performance of computers on the user's network and pose significant security risks to the user's business. Grayware often performs a number of unwanted actions, such as annoying users with pop-up windows, tracking user habits and unnecessarily exposing the computer to attacks.

22.1. Scam

"Scam is a term used to describe a fraudulent scheme or deception in which someone is tricked into giving away money or personal information. Scams can take many different forms, such as phishing scams, investment scams, lottery scams and technical support scams, to name a few.

Phishing scams are attempts to trick people into revealing sensitive information, such as passwords or credit card numbers, by posing as a trustworthy entity. Investment scams persuade people to invest money in a bogus business or financial scheme with the promise of high returns. Lottery scams are messages informing people that they have won a large sum of money in a lottery, but asking them to pay a small fee or provide personal information to claim the prize. Tech support scams are attempts to trick people into paying for unnecessary computer support services by pretending to be from a reputable tech company.

Scammers often use persuasion and urgency to get people to hand over money or personal information. It is important to be wary of unsolicited messages or offers, and to independently verify the legitimacy of any request for personal information or money. You can protect yourself against fraud by being aware of common scams, being wary of unsolicited messages or offers, and never giving out personal information or money without verifying the identity of the recipient.

22.2. Adware

Adware is software that displays advertising banners in web browsers such as Chrome, Internet Explorer and Mozilla Firefox. Although it is not classified as malware, many users find adware invasive. Adware programs often have undesirable effects on a system, such as annoying pop-up ads and general degradation of network connection or system performance. Adware programs are usually installed as separate programs bundled with certain free software from websites. Many users inadvertently agree to install adware by accepting the End User License Agreement (EULA) of the free software. Adware is also often installed together with spyware programs. Both programs benefit from each other's features - spyware programs profile users' Internet behavior, while adware programs display targeted advertisements that correspond to the collected user profile.

22.3. Spyware

Spyware is a type of computer virus that hides on your computer or mobile device, records your private data and sends that information back to whoever created it or monitors it. The tricky thing about spyware, and what separates it from the growing threat of ransomware is the fact that, spyware is designed to both install discretely and operate silently in the background.

Spyware is software that installs components on a computer to record browsing habits (primarily for marketing purposes). Spyware sends this information to its creator or to other interested parties when the computer is online. Spyware is often downloaded along with other components that are referred to as "free downloads" or "freeware" without informing the user about their existence or asking for permission to install them. The information that spyware components collect can include user's keystrokes (keylogging), which means that private information such as login names, passwords and credit card numbers can be stolen. Spyware collects data, such as account names, passwords, credit card numbers and other confidential information, and transmits it to third parties.

22.4. Malvertising

Malvertising, a combination of "malicious" and "advertising", refers to the distribution of malware through online advertising. Cybercriminals use legitimate ad networks to place malicious ads on trusted websites. Users can become infected by clicking on or even just viewing these ads.

How it works

- Inclusion in ad networks: Malicious ads are injected into legitimate networks.
- Spread on websites: These ads appear on popular websites.
- Distribution of malware: Clicking on or viewing the ad can lead to malware infection.

Types of malware

- Ransomware: Encrypts files and holds them for ransom.
- Spyware: Steals confidential information.
- Adware: Displays unwanted advertisements.
- Trojans: Allow unauthorised access to systems.

Protective measures

- Ad blockers: Prevent malicious ads from loading.
- Updated software: Reduces risks from known vulnerabilities.
- Security software: Antivirus programs and firewalls provide protection.
- Be careful what you click: Avoid clicking on suspicious ads.

Chapter 23. Backdoors

A point of access to a hidden program/system. Backdoors are usually intentionally created by a programmer for debugging or maintenance purposes, but if compromised, they can pose a security risk to unauthorized users or software, allowing access and causing damage. Malware often installs Backdoors on compromised systems!

Chapter 24. Botnets

A bot is a programs that run automated tasks over the Internet. Botnets are collection of bots that run autonomously and automatically. Typically they perform repetitive tasks at a much higher rate than a human is capable of. They can be used for malicious purposes, such as denial of service attacks or infecting other computers. An infected computer is called a bot or zombie.

Chapter 25. Macro viruses

A macro is a piece of code that can be embedded in a data file. A macro virus is thus a virus that exists as a macro attached to a data file. In most respects, macro viruses are like all other viruses. The main difference is that they are attached to data files (i.e., documents) rather than executable programs. Document-based viruses are, and will likely continue to be, more prevalent than any other type of virus.

Chapter 26. Worms

Worms are very similar to viruses in that they are computer programs that replicate functional copies of themselves (usually to other computer systems via network connections) and often, but not always, contain some functionality that will interfere with the normal use of a computer or a program. Unlike viruses, however, worms exist as separate entities; they do not attach themselves to other files or programs. Because of their similarity to viruses, worms also are often referred to as viruses.

Chapter 27. Protestware

In March 2022, a developer of **node-ipc** was caught adding malicious code to the popular open source package that deleted files on computers in Russia and Belarus. This was part of a protest that angered many users and raised concerns about the security of free and open source software. The node-ipc update is just one example of what some researchers call **protestware**. Most protest programs related to the Russian invasion of Ukraine simply display anti-war and pro-Ukrainian messages. However, in at least one project, virus-like code was added that aimed to cripple computers in Russia and Belarus. This led to criticism and accusations of causing collateral damage. But there are also examples of protest in the open source scene. Observers of the scene so far found about two dozen software projects that inserted "code against war."

Open-source programs can be modified and viewed by anyone, making them more transparent - and, at least in this case, more vulnerable to sabotage. The protestware event highlights some of the risks that arise when legions of volunteer developers create the code that is critical to running hundreds or thousands of other applications. Some open source software automatically downloads and integrates new versions, and even for those that don't, the vast amount of code often makes manual review infeasible. This means that an update by a single person can mess up an untold number of downstream applications. In that sense, this can be considered a "game changer."

Russia's largest bank has asked its customers to stop updating its software because it is under threat from "protestware". In response to the threat, Russian state-owned bank Sberbank even advised its Russian customers to manually check the source code of the software they need - a security measure that is not feasible for most users. "We urge users to stop updating software and developers to tighten monitoring when using external code," Sberbank said, according to Russian media and cybersecurity firms.

Chapter 28. Stealth viruses

What is a stealth virus? A stealth virus is one that, while active, hides the modifications it has made to files or boot records. It usually achieves this by monitoring the system functions used to read files or sectors from storage media and forging the results of calls to such functions. This means that programs that try to read infected files or sectors see the original, uninfected form instead of the actual, infected form. Thus the virus's modifications may go undetected by antivirus programs. However, in order to do this, the virus must be resident in memory when the antivirus program is executed, and the antivirus program may be able to detect its presence.

The very first DOS virus, Brain, a boot-sector infector for example monitored physical disk input/output and redirected any attempt to read a Brain-infected boot sector to the disk area where the original boot sector was stored.

28.1. File stealth viruses

In addition to hiding the boot information, DOS file stealth viruses attack .com and .exe files when opened or copied, and hide the file size changes from the DIR command. The major problem arises when you try to use the CHKDSK/F command and there appears to be a difference in the reported files size and the apparent size. CHKDSK assumes this is the result of some cross-linked files and attempts to repair the damage. The result is the destruction of the files involved.

28.2. Full stealth viruses

With a full stealth virus, all normal calls to file locations are cached, while the virus subtracts its own length so that the system appears clean.

28.3. Countermeasures against Stealth Viruses?

You need a clean system so that no virus is present to distort the results of system status checks. Thus you should start the system from a trusted, clean, bootable diskette before you attempt any virus checking.

Chapter 29. Encryption

One method of evading malware detection is to use simple encryption to encipher (encode) the body of the malware, leaving only the encryption module and a static cryptographic key in clear text which does not change from one infection to the next.

29.1. What is a polymorphic virus?

A polymorphic virus is one that produces varied but operational copies of itself. This strategy assumes that virus scanners will not be able to detect all instances of the virus. One method of evading scan-string driven virus detectors is self-encryption with a variable key. Polymorphic code was the first technique that posed a serious threat to virus scanners.

More sophisticated polymorphic viruses (e.g., V2P6) vary the sequences of instructions in their variants by interspersing the decryption instructions with "noise" instructions (e.g., a No OPeration instruction (NOP), or an instruction to load a currently unused register with an arbitrary value), by interchanging mutually independent instructions, or even by using various instruction sequences with identical net effects (e.g., Subtract A from A, and Move 0 to A). A simple-minded, scan-string based virus scanner would not be able to reliably identify all variants of this sort of virus; in this case, a sophisticated scanning engine has to be constructed after thorough research into the particular virus.

One of the most sophisticated forms of polymorphism used so far is the Mutation Engine (MtE) or the Trident Polymorph Engine (TPE), which comes in the form of an object module. With such mutation engines, any virus can be made polymorphic by adding certain (API) calls to its assembler source code and linking to the mutation-engine and provided random-number generator modules.

The advent of polymorphic viruses has rendered virus scanning an increasingly difficult and expensive endeavor; adding more and more search strings to simple scanners will not adequately deal with these viruses.

Chapter 30. What is an armored virus?

Armored viruses use special tricks to make the tracing, disassembling, and understanding of their code more difficult. A good example is the Whale virus. An armored virus uses various techniques to evade detection, such as encrypting its code, obfuscating its code, and using anti-debugging and anti-tampering methods.

Armored viruses pose a serious threat because they can be used to perform malicious activities such as stealing sensitive information, altering or corrupting data, and slowing performance without being detected. They can also be used as part of more complex attacks, such as advanced persistent threats (APTs), to maintain a foothold on a target network over an extended period of time.

Chapter 31. What is Phishing/Vishing?

Phishing and vishing are types of scams used to steal sensitive information such as passwords, credit card numbers and other personal data.

Phishing is a type of scam that tricks people into providing sensitive information through fake emails or websites that appear to be from a reputable source, such as a bank or a well-known company. The goal of phishing scams is to trick people into revealing personal information, such as passwords or credit card numbers, by posing as a trustworthy entity.

Vishing, short for voice phishing, is a type of phishing scam where people are tricked into revealing sensitive information over the phone. In vishing scams, scammers often pretend to be from a bank, government agency or technology company and use persuasive techniques to get people to reveal sensitive information.

Both phishing and vishing scams are becoming increasingly sophisticated and it is important to be wary of unsolicited emails or phone calls. To protect yourself from these types of scams, never provide sensitive information in response to an unsolicited request and independently verify the identity of the recipient before providing any personal information.

Chapter 32. Secure Boot/UEFI/Firmware Malware

In 2012, an industry-wide coalition of hardware and software makers adopted Secure Boot as a crucial defense mechanism against a growing and sophisticated security threat: malware targeting the system's firmware, specifically the BIOS. The BIOS, or Basic Input/Output System, is the firmware responsible for initializing hardware components and loading the operating system every time a computer is powered on. Malware that infects the BIOS poses an especially insidious threat because it can establish a foothold deep within the system, evading traditional detection methods and persisting through operating system reinstalls. Once entrenched, such malware can execute before the operating system and any security software, making it extremely difficult to detect and remove.

The threat of BIOS-dwelling malware had long been considered theoretical, heightened by the creation of the ICLord BIOS rootkit by a Chinese researcher in 2007. ICLord was a proof-of-concept rootkit, a type of malware designed to gain and maintain privileged access to a system while remaining hidden from standard security measures. This proof of concept not only demonstrated the feasibility of BIOS rootkits but also underscored their potential power. While ICLord remained a theoretical threat, it set the stage for the realization of more dangerous malware.

In 2011, the landscape of firmware security changed dramatically with the discovery of Mebromi, the first-known BIOS rootkit to be observed in the wild. Mebromi marked the transition from theoretical to actual threat, underscoring the vulnerability of BIOS firmware to sophisticated attacks. Mebromi was capable of infecting the BIOS, overwriting it with malicious code, and maintaining persistence even after the operating system was reinstalled—a stark reminder of the critical need for stronger security measures.

Recognizing the severity of the threat posed by Mebromi and other potential firmware attacks, the architects of Secure Boot developed a sophisticated security framework aimed at fortifying the pre-boot environment. Integrated into the Unified Extensible Firmware Interface (UEFI)—which was designed to replace the aging BIOS system—Secure Boot leverages public-key cryptography to verify the integrity and authenticity of firmware and software components before they are loaded. Specifically, Secure Boot only allows the execution of code that is signed with a recognized and trusted digital signature, effectively preventing unauthorized or malicious code from compromising the system at such an early and vulnerable stage.

To this day, Secure Boot is regarded by security experts and organizations—including Microsoft and the US National Security Agency—as a foundational element in protecting devices, particularly in critical environments such as industrial control systems and enterprise networks. Its role in establishing a chain of trust from the hardware through to the operating system is considered essential in defending against the sophisticated and persistent malware threats that continue to evolve. The adoption of Secure Boot represents a significant milestone in the ongoing effort to enhance cybersecurity and protect against the increasingly complex landscape of malware attacks.

In 2024 Secure Boot is considered to be broken because cryptografic keys to protect secure boot were leaked (PKfail)

32.1. UEFI Bootkits in General

UEFI bootkits are a type of malware that hides in the UEFI firmware of a computer. UEFI is a software program that controls the boot process of boot process and execute malicious code before the operating system even starts.

- UEFI bootkits are difficult to detect and remove because they hide in the UEFI firmware.
- UEFI bootkits can be used to install persistent malware that remains active even after the operating system is reinstalled.
- UEFI bootkits can be used to steal sensitive data, such as passwords and bank details.

Countermeasures:

- Enable UEFI Secure Boot to prevent unauthorized code from running during the boot process.
- Keep your system firmware and operating system up to date to patch vulnerabilities.
- Use reliable antivirus software to detect and remove UEFI bootkits.

32.2. Rootkits and Bootkits

Rootkits and bootkits are sophisticated forms of malware that compromise system integrity by gaining unauthorized control over devices. Operating at low levels within an operating system or firmware, they are particularly challenging to detect and remove. This document outlines their functionality, types, techniques, and strategies for mitigation.

- Rootkits are tools that allow attackers to gain and maintain privileged access to a system while concealing their presence.
- Bootkits are a specific type of rootkit that infects the boot process, compromising systems before the operating system loads.

Both rootkits and bootkits pose significant security risks due to their stealthy operations and privileged access capabilities. Both rootkits and bootkits often conceal their presence by altering system internals. Traditional antivirus solutions may struggle to detect them due to their low-level operations.

Rootkits and bootkits represent significant threats to modern computing environments due to their stealthy nature and privileged access capabilities. Effective detection and mitigation require robust security practices, including hardware-based protections, monitoring tools, and proactive patching. Understanding these threats is essential for safeguarding critical systems and sensitive data.

Chapter 33. Links / Pointers

https://en.wikipedia.org/wiki/Timeline_of_computer_viruses_and_worms

Chapter 34. Some very old (DOS) viruses that were very widespread in the past

MS-DOS viruses were particularly prevalent during the early days of personal computing, exploiting the relatively limited security measures of that era. Here are some notable MS-DOS viruses:

34.1. CIH (Chernobyl)

- **Origin**: Taiwan, 1998
- **Type**: File virus
- **Description**: Known for its destructive payload that triggered on April 26, it could overwrite critical parts of the BIOS, rendering computers unbootable. Although not an MS-DOS virus, it affected Windows 9x systems that were based on DOS.

34.2. Sasser

- **Origin**: Germany, 2004
- **Type**: Worm
- **Description**: While not strictly an MS-DOS virus, Sasser exploited vulnerabilities in Windows systems to spread. It caused widespread disruptions in the early 2000s.

34.3. Melissa

- **Origin**: USA, 1999
- **Type**: Macro virus
- **Description**: This virus spread through Microsoft Word documents and email, causing substantial email server disruptions. While primarily a macro virus, its impact was significant across various Windows environments.

34.4. Lehigh

- **Origin**: USA, 1987
- **Type**: Boot sector virus
- **Description**: One of the early boot sector viruses, it specifically targeted the master boot record and could corrupt the entire hard disk.

34.5. Form

- **Origin**: Probably Swiss, early 1990s
- **Type**: Boot sector virus

- **Description**: This virus became widely known for its payload that activated on the 18th of each month, causing the keyboard to behave erratically.

34.6. Elk Cloner

- **Origin**: USA, 1982

- **Type**: Boot sector virus (on Apple II, not MS-DOS, but historically significant)

- **Description**: One of the earliest known viruses, it displayed a poem on the 50th boot of an infected system. Although not an MS-DOS virus, it is significant in the history of computer viruses.

34.7. Ping Pong (Bouncing Ball)

- **Origin**: Italy, 1988

- **Type**: Boot sector virus

- **Description**: This virus caused a bouncing ball effect on the screen and infected the boot sector of floppy disks.

These viruses highlight the diverse strategies employed by malware developers in the MS-DOS era, from boot sector infections to file-based and polymorphic techniques, illustrating the early challenges of computer security.

34.8. The Brain Virus: The Birth of the Computer Virus Era

The Brain virus, often cited as the first IBM PC-compatible virus, marked a significant milestone in the history of computer security. Created in 1986 by two brothers in Pakistan, it initiated an era of growing cybersecurity threats and responses.

The Brain virus was developed by Basit and Amjad Farooq Alvi, who operated a computer store in Lahore, Pakistan. Frustrated with the piracy of their medical software, they created the virus as a form of copy protection, embedding their contact information within the code to raise awareness about piracy.

The Brain virus serves as a historical landmark in cybersecurity, highlighting the early challenges of digital security and the unintended consequences of technological interventions. It underscored the need for ongoing vigilance and education in the evolving landscape of cybersecurity.

34.8.1. Technical Details

The Brain virus is a boot sector virus, infecting the boot sector of storage media like floppy disks. It becomes resident in memory when the computer boots from an infected disk and then infects any clean disks accessed by the system. It displays a message with the authors' names and contact details, an unusual feature among viruses.

Infection Mechanism:

- **Boot Process**: Loads into memory during boot from an infected disk.

- **Replication**: Spreads to other disks accessed by the infected computer.

- **Infected Message**: Displays a message with the creators' contact information.

Attributes:

- **Memory Resident**: Remains active until the computer is turned off.

- **Stealth Techniques**: Hides its presence by intercepting system calls.

- **Non-Destructive**: Does not delete or corrupt files, focusing instead on spreading and delivering a message.

34.8.2. Impact and Spread

The Brain virus's impact was significant, as it was the first widely recognized PC boot virus. It spread rapidly through the common practice of sharing floppy disks, reaching users worldwide by the late 1980s.

Geographical Spread:

- **Local to Global**: Initially spread within Pakistan, then globally through shared software.

Economic and Social Impact:

- **Awareness and Fear**: Raised awareness about computer security and vulnerabilities.

- **Economic Consequences**: Prompted investments in antivirus software and improved security practices.

34.9. Cascade

Cascade virus (also known as Herbstlaub in Germany) is a well-known DOS computer virus that is a memory-resident virus written in assembly language. Cascade was widely spread in the 1980s and early 1990s. It infected DOS .COM files and caused the text on the screen to cascade down and form a pile at the bottom of the screen. It was notable for the fact that it used an encryption algorithm to avoid detection. However, it could be seen that the size of the infected files increased by 1701 or 1704 bytes. In response, IBM developed its own anti-virus software.

When a file infected with Cascade is introduced into a system and executed, the virus checks the BIOS for the string "COPR. IBM", an IBM copyright notice in the BIOS. If it finds the string, it tries to stop there, but fails, and the virus becomes memory resident. Every time a .COM file is executed, the virus starts infecting it. It replaces the first three bytes of the new host file with code that references the virus code. The virus places the original first three bytes of the host into its own code.

Cascade's payload is executed when an infected file is executed between October 1 and December 31, 1988. It causes characters on a DOS screen to randomly drop down in a pile of numbers and letters. Variants can also cause noise.

The virus has a number of variants. Cascade-17Y4, which is believed to have originated in

Yugoslavia, is almost identical to the most common 1704-byte variant. One byte has been changed, probably by a random "mutation". However, this has resulted in a "bug" in the virus. Another mutated variant is also known - it infects the same file over and over again.

34.10. Jerusalem

Jerusalem is a DOS virus which was first detected in Jerusalem in October 1987. Its origin is uncertain, as it was thought to have originated in Israel, but evidence from 1991 suggests that it may have originated in Italy. As of 1993, Jerusalem was still in the wild and many variants had been created. The last reported case of Jerusalem was in 1995, almost 8 years after its discovery. The virus has gone by many names, some referring to its possible origin and its Friday the 13th payload date. Jerusalem was initially very common (for a virus at the time) and spawned a large number of variants. However, since the advent of Windows, these DOS interrupts are no longer used, so Jerusalem and its variants have become obsolete.

Once infected, the Jerusalem virus becomes memory resident (using 2kb of memory) and then infects every executable file that is run, except for COMMAND.COM. COM files grow by 1,813 bytes when infected by Jerusalem and are not re-infected. EXE files grow between 1,808 and 1,823 bytes each time they are infected. The virus re-infects .EXE files each time they are loaded until they are too large to load into memory. Some .EXE files are infected but do not grow because multiple overlays follow the real .EXE file in the same file. Sometimes .EXE files are infected by mistake, so that the programme fails to run when it is run.

The virus code itself hooks into interrupt processing and other low level DOS services. For example, code in the virus suppresses the printing of console messages if, for example, the virus is not able to infect a file on a read-only device such as a floppy disk. One of the clues that a computer is infected is the mis-capitalization of the well-known message "Bad command or file name" as "Bad Command or file name".

The program contains one destructive payload that is set to go off on Friday the 13th, all years but not in 1987. On that date, the virus deletes every program file that was executed. Jerusalem is also known as BlackBox because of a black box it displays during the payload sequence. If the system is in text mode, Jerusalem creates a small black rectangle from row 5, column 5 to row 16, column 16. The rectangle is scrolled up by two lines.

As a result of the virus hooking into the low-level timer interrupt, PC-XT systems slow down to one fifth of their normal speeds 30 minutes after the virus has installed itself. The slowdown is less noticeable on faster machines. The virus contains code that enters a processing loop each time the processor's timer tick is activated.

Symptoms also include spontaneous disconnection of workstations from networks and creation of large printer spooling files. Disconnections occur since Jerusalem uses the 'interrupt 21h' low-level DOS functions that Novell Netware and other networking implementations required to hook into the file system.

Variants

Over the years that Jerusalem spread, many virus coders created variants of the virus, making Jerusalem one of the largest families of viruses ever created. It even includes many sub-variants and a few sub-sub-variants. Most variants are unimaginative, simply changing the payload date,

text displayed or even nothing at all. Some variants contain fixes for the bugs of the original.

```
Jerusalem.1013 Jerusalem.1024 Jerusalem.1234 Jerusalem.1237 Jerusalem.1238
Jerusalem.1241 Jerusalem.1244 Jerusalem.1264 Jerusalem.1291 Jerusalem.1329
Jerusalem.1347 Jerusalem.1348 Jerusalem.1349 Jerusalem.1353 Jerusalem.1356
Jerusalem.1361 Jerusalem.1363 Jerusalem.1364 Jerusalem.1390 Jerusalem.1399
Jerusalem.1408 Jerusalem.1427 Jerusalem.1446 Jerusalem.1448 Jerusalem.1455
Jerusalem.1459 Jerusalem.1477 Jerusalem.1478 Jerusalem.1487 Jerusalem.1488
Jerusalem.1489 Jerusalem.1500 Jerusalem.1503 Jerusalem.1504 Jerusalem.1511
Jerusalem.1518 Jerusalem.1521 Jerusalem.1522 Jerusalem.1523 Jerusalem.1524
Jerusalem.1525 Jerusalem.1526 Jerusalem.1530 Jerusalem.1533 Jerusalem.1536
Jerusalem.1548 Jerusalem.1552 Jerusalem.1558 Jerusalem.1562 Jerusalem.1568
Jerusalem.1570 Jerusalem.1587 Jerusalem.1589 Jerusalem.1591 Jerusalem.1596
Jerusalem.1598 Jerusalem.1605 Jerusalem.1607 Jerusalem.1624 Jerusalem.1631
Jerusalem.1639 Jerusalem.1640 Jerusalem.1653 Jerusalem.1664 Jerusalem.1682
Jerusalem.1692 Jerusalem.1715 Jerusalem.1716 Jerusalem.1720 Jerusalem.1721
Jerusalem.1728 Jerusalem.1733 Jerusalem.1735 Jerusalem.1747 Jerusalem.1756
Jerusalem.1765 Jerusalem.1767 Jerusalem.1768 Jerusalem.1783 Jerusalem.1792
Jerusalem.1807 Jerusalem.1808 Jerusalem.1813 Jerusalem.1824 Jerusalem.1845
Jerusalem.1884 Jerusalem.1888 Jerusalem.1899 Jerusalem.1960 Jerusalem.1968
Jerusalem.1970 Jerusalem.1975 Jerusalem.1984 Jerusalem.1991 Jerusalem.2000
Jerusalem.2012 Jerusalem.2027 Jerusalem.2053 Jerusalem.2064 Jerusalem.2080
Jerusalem.2082 Jerusalem.2083 Jerusalem.2116 Jerusalem.2126 Jerusalem.2128
Jerusalem.2132 Jerusalem.2187 Jerusalem.2208 Jerusalem.2223 Jerusalem.2224
Jerusalem.2272 Jerusalem.2291 Jerusalem.2350 Jerusalem.2358 Jerusalem.2368
Jerusalem.2389 Jerusalem.2437 Jerusalem.2465 Jerusalem.2472 Jerusalem.2490
Jerusalem.2576 Jerusalem.2758 Jerusalem.2880 Jerusalem.2886 Jerusalem.3887
Jerusalem.4112 Jerusalem.5120 Jerusalem.641 Jerusalem.662 Jerusalem.679
Jerusalem.878 Jerusalem.880 Jerusalem.986 Jerusalem.A Jerusalem.CVEX
Jerusalem.Curse Jerusalem.June11_T3Scan Jerusalem.Plastique
Jerusalem.Roger Jerusalem.a Jerusalem.com Jerusalem.sURIV_3
```

34.11. The Tequila Virus

It was one of the early examples of a polymorphic virus, making it harder to detect by antivirus software because it changed its code each time it infected a new file. The Tequila virus, originating in Switzerland in 1991, is one of the earliest examples of polymorphic malware. This advanced virus significantly impacted the evolution of malware and cybersecurity strategies by evading detection through code mutation.

34.11.1. Technical Details

The Tequila virus infected executable (.EXE) files on DOS-based systems. It was a memory-resident virus that used polymorphism to alter its code with each new infection, making it difficult for signature-based antivirus programs to detect it.

Infection Mechanism:

1. **File Infection**: Attaches to executable files, embedding its encrypted code.

2. **Memory Residency**: Becomes resident in memory upon execution of an infected file, allowing it to infect other executables.

3. **Polymorphism**: Uses a mutation engine to change its code with each infection, evading detection.

Technical Attributes:

- **Encryption**: Hides its presence within infected files using encryption.

- **Mutation Engine**: Alters its code with each infection, complicating detection.

- **Payload**: Displays a fractal, alerting users to its presence.

Tequila pioneered polymorphic techniques, influencing future malware development and antivirus strategies. The Tequila virus was a landmark in the history of computer viruses, demonstrating sophisticated polymorphic techniques that challenged and shaped the development of antivirus technologies and cybersecurity strategies.

34.12. Stoned

The Stoned Virus Family: A Legacy of Boot Sector Infections

Stoned is a very large family of boot sector viruses on the DOS platform, originating in early 1988. This family of viruses became infamous for its persistent and insidious nature, with notable members such as the Michelangelo virus, which incited widespread panic in the early 1990s, and the Angelina virus from 1994, which notably resurfaced on infected laptops as late as 2007. The Stoned virus was allegedly created by a student at the University of Wellington in New Zealand.

34.12.1. Mechanism of Infection

When a computer boots from an infected hard drive, Stoned becomes resident in memory, establishing a foothold in the system. If the computer boots from a disk other than the hard disk, Stoned checks the master boot record (MBR) of the hard disk and infects it if it is clean.

- **Floppy Disk Infection**: When infecting a floppy disk, Stoned relocates the master boot record to sector 11 and places its own code in sector 0.

- **Hard Disk Infection**: When infecting the hard disk, it moves the MBR to page 0, cylinder 0, sector 7, and places itself in page 0, cylinder 0, sector 1.

The original variant of the Stoned virus targets only 360-kilobyte 5.25-inch floppy disks and hard disks.

Once resident in memory, Stoned infects the MBRs of all accessed floppy disks but does not reinfect the hard disk. Even if the virus is removed from the MBR while it is still in memory, it does not attempt to reinfect the hard disk.

34.12.2. Payload and Effects

The Stoned virus has a 1 in 8 chance of releasing its payload during the boot process. When this happens, the infected computer emits a beep and displays the following message:

```
Your PC is now stoned! LEGALIZE MARIJUANA!
```

34.12.3. Prominent Members

1. **Michelangelo Virus**: This variant caused significant panic in the early 1990s. Despite the media hype, it infected only a few thousand computers, demonstrating how fear and misinformation can amplify perceived threats.

2. **Angelina Virus**: Identified in 1994, this variant became notable again in 2007 when it was discovered on laptops that had been sold with pre-existing infections. This resurgence highlighted the enduring nature of some viruses and the importance of rigorous cybersecurity measures.

34.12.4. Legacy and Impact

The Stoned virus family exemplifies the early challenges of cybersecurity and the evolving nature of computer threats. Despite its relatively simple mechanics by today's standards, the impact of the Stoned virus family was profound, illustrating the vulnerabilities in early computing systems and the significant role of media in shaping public perception of cybersecurity threats.

34.13. Michelangelo

Michelangelo Virus: A Case of Media-Driven Panic

The Michelangelo virus, which stems from the Stoned boot virus family, is renowned for being one of the first computer viruses to garner widespread media attention. While it incited substantial panic, the actual damage it inflicted was minimal. The virus infected only a few thousand computers, making it a classic example of media-induced hysteria.

The media frenzy began in January 1992 when two coincidental events sparked interest. One computer manufacturer inadvertently shipped 500 computers infected with the Michelangelo virus. On the same day, another manufacturer announced it would begin shipping computers with pre-installed antivirus software. This concurrence captured the media's attention, leading to a flurry of sensationalist reporting.

United Press International played a pivotal role in escalating the panic by interviewing key figures in the cybersecurity field. Among them was the International Partnership Against Computer Terrorism and John McAfee, the president of a prominent antivirus company. These interviews fueled fears, with projections suggesting that hundreds of thousands of computers could be destroyed by the virus. Data recovery consultant Martin Tibor further amplified concerns with dramatic statements like "I find virus disasters everywhere" and "I see victims of viruses all the time."
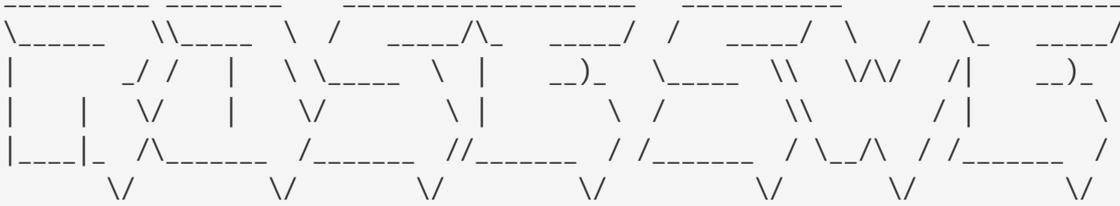
In the weeks leading up to the virus's activation date, the media focused heavily on the potential local impact. Some outlets chose to report more on the growing hysteria than on factual information about the virus itself. Few took the initiative to consult with real experts to mitigate the panic. As a result, a significant number of computer users rushed to purchase antivirus software, driven by the fear of imminent digital disaster.

In hindsight, the Michelangelo virus serves as a poignant reminder of the power of media in shaping public perception and the importance of critical evaluation of such reports. While the virus itself was relatively harmless, the surrounding hype created a disproportionate sense of urgency and fear among computer users.

/* End of Document */

Chapter 35. Copyright

©opyright by

```
 _____ _____    _____  _____     _____
 _____    \\_____   \ /    ____/\_    ____/ /    ____/  \   /  \_    ____/
 |       _/ /   |   \ \_____  \ |     __)_    \_____  \\    \/\/    /|     __)_
 |    |   \/    |    \/       \ |        \   /        \\           /|          \
 |____|_   /_____  /_____  //_____   / /_____   / \__/\  / /_____   /
        \/          \/        \/          \/          \/          \/           \/
```

```
-------------------------------------=-------------------------------------
  ROSE SWE                           See ROSEBBS.TXT for
  Dipl.-Ing. Ralph Roth             full address, FAX and PGP keys.
  http://rose.rult.at
  rose_swe@hotmail.com               All Rights Reserved!
-------------------------------------=-------------------------------------
```

Chapter 36. End

End of the documentation! Thank you for reading it. Bye!