June 1990 ISSN 0956-9979

VIRUS BULLETIN

THE AUTHORITATIVE INTERNATIONAL PUBLICATION
ON COMPUTER VIRUS PREVENTION,
RECOGNITION AND REMOVAL

CONTENTS

EDITORIAL

Computer Viruses as Weapon Systems 2

PROCEDURES

Training and Awareness 3

FEATURE ARTICLE

The Bulgarian Computer Viruses - 'The Virus Factory' 6

ANTI-VIRUS SOFTWARE FOR IBM PCs 10

KNOWN IBM PC VIRUSES (UPDATE) 11

TOOLS & TECHNIQUES

Dynamic Decompression, LZEXE and the Virus Problem 12

WORM PROGRAMS

The Internet Worm - Action and Reaction 13

PRODUCT REVIEW

Certus - Tools To Fight Viruses With 18

END-NOTES & NEWS 20

EDITORIAL

Computer Viruses as Weapons Systems

There has been some speculation recently about military research into the use of virus programs to disrupt enemy computer and communications systems. Military electronic countermeasures (ECM) might, it has been argued, encompass computer viruses, worms and other forms of replicating attack programs. This speculation is almost impossible to substantiate. In the UK for instance, electronic intelligence (ELINT) and signals intelligence (SIGINT) are highly classified fields and no information about research and development projects is made publicly available.

Interception of enemy communications and cipher-breaking are the primary SIGINT and ELINT functions. Recent military computer security efforts have attempted to counter 'leakage' - the capacity for enemy intelligence to intercept information processed by computers. Leakage can occur through electromagnetic radiation emanating from terminals, processors and cabling and this has led to the development of TEMPEST shielding designed to confound eavesdropping. Another possible cause of leakage is the Trojan horse or 'sleeper' program designed to exploit covert channels and assist illicit communication to unauthorised parties. Viruses, worms and Trojans can be used to exploit covert channels and incoming software for high security computing systems is usually supplied as source code for prior analysis. Covert channels are discussed in the US *Department of Defense Trusted Computer System Evaluation Criteria* - the 'Orange Book' - and are a major consideration in the design of secure computer systems. *

Offensive Applications

Securing information is a defensive measure. Military agencies are equally interested in offensive operations. This might involve intercepting enemy communications and information and, if necessary, destroying or disrupting those facilities. The following information appeared recently on the *Virus-L* conference:

> The US Department of Defense recently published a booklet titled PROGRAM SOLICITATION 90.2 FY-1990 SMALL BUSINESS INNOVATION RESEARCH (SBIR) PROGRAM. Page 45 contains project A90-217 'Computer Virus Electronic Countermeasure (ECM)'. This project, for 'exploratory development' is a feasibility study to determine:
>
> "The potential for using 'computer viruses' as an ECM technique against generic military communications systems/nets. The goal shall be to determine the feasibility of remotely introducing a virus into a system/net and analyzing its effects on various subsystem components.
>
> DESCRIPTION: The purpose of this research shall be to investigate potential use of computer viruses to achieve traditional communications ECM effects in targeted communications systems. These effects can include data (information) disruption, denial, and deception, but other effects should also be researched such as executable code in processors, memory storage management, etc.
>
> Research in effective methods or strategies to remotely introduce such viruses should be conducted. Efforts in this area should be focused on RF atmospheric signal transmission such as performed in tactical military data communications."

For some extraordinary reason, this project has been 'farmed out' to private enterprise. Jim Vavrina, a computer security specialist with the US *Army Information Systems Software Centre* who uploaded the message, says: "Admittedly, myself and my colleagues are quite surprised that something of this nature would be put on the streets for research and not use the expertise internally available."

Program Solicitation 90.2 might lead us to conclude that the US Department of Defense has 1) *been very slow to appreciate the virus threat and its implications for security and electronic warfare* and 2) *regards the issue rather casually and believes it unnecessary to use its immense 'in-house' resources for research and development purposes.*

Alternatively, Program Solicitation 90.2 could be regarded as a classic example of disinformation to conceal the true extent of US military virus research.

Morris Sentenced

Robert Morris has avoided a prison sentence for attacking the US DARPA Internet network. In early May a Federal judge placed him on three year's probation, ordered him to complete 400 hours of community service, and imposed a fine of $10,000 payable within twelve months.

An interesting aspect of the case was the estimated level of damage (assessed in US dollars) caused by the program. A report from John McAfee of the *Computer Virus Industry Association* claimed that the Internet worm had caused a staggering $96 million dollars of damage. This report was dismissed as "grossly exaggerated and self-serving" by the Cornell commission of enquiry into the incident headed by Mr. M Stuart Lynn. A figure of $200,000 quoted by Gene Spafford of the *University of Purdue* has been accepted as a realistic estimate of the damage inflicted by most knowledgeable observers.

As *VB* reported in April, hackers are still penetrating Internet and according to some reports the network's security is as lax as it was before the Morris program struck. There is still concern that known bugs and trapdoors in this and other wide area networks remain unfixed, that issued 'patches' are being ignored by systems administrators and that manufacturers and vendors are failing to address network security. In the UK, the Joint Academic Network (JANET) remains particularly vulnerable to security breaches with little concerted effort being made to 'plug the gaps'.

The Morris sentence (widely regarded as lenient) will probably do little to deter network pranksters, either in the US or elsewhere. Under the circumstances a repeat of the Internet worm, or similar, seems inevitable.

** Trojan horses and covert channels will be the subject of a commissioned article to appear in VB, volume II.*

PROCEDURES

Training and Awareness

Computer viruses are created by people and are spread by people. There are no purely technical solutions to the virus problem because productivity depends upon people using computers and software. Unless floppy disk drives are disabled, the user will always be able to install software and execute it.

Thus, there is no point in implementing technical security measures if a) *people will not use them* and b) *people fail to understand why particular measures are necessary.*

Good management depends upon clear and straightforward explanation. If new procedures are implemented it is essential to explain the reasons for them and to outline peoples' responsibilities. **The computer virus problem is too complex to allow uninformed employees to make on-the-spot decisions.** Rules are necessary and, if presented and explained clearly, will be welcomed by employees.

Authorised Software

It is important to forbid the installation and/or execution of unauthorised software. In working environments this often means games and demo disks, although users should also be cautious of unfamiliar programs (which may have enticing or intriguing names such as SEX.COM) already installed on the machine.

The aim should be to encourage the user to submit software for inspection by an 'in-house' specialist or team. **The establishment of a small, technically competent group to vet all incoming software will significantly increase control and reduce the risks from replicating malicious software.** This group will have specialist knowledge about computer virus detection and contingency plans to deal with infection (*See Software Quality Assurance Section (SQAS), VB, May 1990*).

However, users will only comply with such a rule if they understand the reasons behind it. This involves explaining what a computer virus is and how it infects computers and software. A presentation by an informed member of staff will help to solve this problem. **Do not assume knowledge on the part of the user.** The average PC user does not know what a boot sector is, let alone an interrupt vector! Remember, also, that this material must be presented to everyone - senior management are as likely to be as ignorant about this particular subject as the most junior staff (possibly more so).

What Should Be Explained?

This depends upon the computing experience of the audience. The most basic presentation might include the following points:

1. A computer virus (never use the term 'data virus') is a program that modifies normal programs to include a copy of itself. Viruses also spread from one computer to another. This can happen on a network or with stand-alone computers. Computer viruses are created by people to damage and disrupt computer systems. (*Depending on the level of user knowledge, it may be necessary to distinguish between programs and data. Emphasise that pure data cannot spread viruses*).

2. Viruses are particularly dangerous because they do not reveal their presence until they 'trigger'. After triggering, some viruses destroy everything stored on the computer.

3. Programs (software) can spread viruses. In most office environments this means floppy disks. (*Decide which media used by a particular audience could spread a virus (see VB, May 1990) and display them. Company policy on exchanging software should be clearly outlined*).

4. Clearing a computer or network of a virus infection is a time-consuming, expensive and very disruptive business. In the event of a destructive virus triggering, hundreds of man-hours of data entry could be lost. This will damage the company and its employees.

5. Viruses are a common and serious problem and for this reason employees must submit all software which they wish to use for inspection prior to installing it. (*Designate an individual or team to which users can report. Specify a name (Mr. Smith), a location (Rm 123) and a telephone number (Ext 4137).*

6. If machines have no hard disk, tell users to always boot from a write-protected system floppy disk and nothing else. It is helpful to identify system disks using coloured adhesive labels or using distinctively coloured diskettes.

By adopting this method of education, management gain far greater control over the software in use and, most importantly, remove the anti-virus burden from non-technical staff.

If time or resources are not available to establish a Software Quality Assurance Section, the role of 'PC hygiene' must be emphasised (see below).

A Little Knowledge is a Dangerous Thing

In teaching users about malicious programs, avoid revealing information which could be misused. There is no need, for instance to explain a DOS FORMAT command, if such a command is irrelevant to the user.

It is also probable that having described viruses, how they work and what they do, the DP department will have to respond to false alarms from users misinterpreting normal machine behaviour.

It is therefore advisable to outline unusual symptoms which should be reported. These include:

- **Strange graphics or text**
- **Alteration to text or commands**
- **Unexpected sound effects**
- **Degraded system performance**
- **Unexpected disk access**
- **Changes in file lengths/date/time/stamps**
- **Bad sectors on floppy disks/increased number of bad sectors on hard disk**
- **Reduced memory**
- **Unknown files or directories appearing**
- **Alterations to the system clock**
- **Software failure/slow program execution**
- **Unusual behaviour after rebooting/refusal to reboot**

A word of caution; do not depend upon users to accurately diagnose and/or report strange machine behaviour. Diagnostic tools and anti-virus software is designed for this purpose and is far better at it.

User Awareness

Accompanying the presentations, should be a campaign to increase awareness amongst the workforce. This entails a mixture of commonsense and the use of leaflets, posters, and other training materials. The military maxim KISS (Keep It Simple Stupid) is essential to any general information campaign. The message will not sink in if it is presented in an overly technical manner. Equally, it is important to avoid sensationalism. Factual, straightforward material is best.

Policies

Management should prepare a company plan which addresses the following issues:

***** The use of shareware, public domain software, bulletin boards, games and demo disks. Each should be regarded as a separate category and the policy devised accordingly.

***** The use of home computers for company work.

***** Responsibility for backups, software and data archives. Decide who will be responsible for taking PC backups and how this material will be indexed and stored safely.

***** Anti-virus software. This involves identifying critical systems and determining the exact security requirements for machines, media and networks. It is important to establish an evaluation protocol to assess the effectiveness and suitability of products. Staff should be designated to maintain and control anti-virus software.

***** Reporting a virus infection or other incident. The chain of command should be decided and specialist technical staff identified.

***** Controlling a virus attack. The personnel and procedures to be involved, the circumstances under which outside help should be sought and the sources of corrective software should be identified.

***** The exact procedure for diagnosing and isolating infected disks and PCs.

***** Recovery from infection to include backup and dump policies and damage assessment.

***** The use of legal and/or disciplinary procedures.

***** Education and training. Management should decide the types of training provided to employees and the methods used to educate them. It is also important to establish 'rumour control' - the public relations procedures necessary to prevent unauthorised leaks of information to the press or others.

Policy Documents and the Contingency Plan

Once policy is decided it should be formalised and documented as a company plan which should be distributed to relevant technical and general management.

A booklet or pamphlet explaining policy and procedures on microcomputer security should be distributed to **all** employees. This must be easy to read and understand. The same rules and procedures should apply throughout the organisation. **The aim of the booklet is prevention; that is stopping malicious software from being installed or executed.**

The contingency plan has a different function. Its objective is to provide a written checklist for containment and recovery from an infection should it occur. This plan will be used by the specialist team(s) designated to deal with computer virus attacks. The contingency plan should be flexible enough to address different types and intensities of infection, but concrete enough to enable a rapid response.

It is vitally important that the contingency plan is formalised as a document. This will ensure that procedures are followed exactly and that nothing is overlooked during a crisis. **The contingency plan should be tested.**

PC Hygiene

The observance of *PC hygiene* is all-important in preventing computer viruses. It is useful to explain the principles of PC hygiene to all employees, particularly if you do not implement 'in-house' software quality assurance.

A virus has little chance of reaching a computer if that computer is not networked, has a limited number of users (preferably only one), and is never used with disks from other sources. **It is therefore important to spell out the potential risks associated with different magnetic media** (*see Infective Media & Routes of Infection, VB, May, 1990).*

The rules are fairly straightforward - **every executable item run on a computer is capable of spreading a virus.** High risk software includes free demonstration disks, shareware, public domain software and bulletin board software. Many large organisations have now forbidden the use of all of these software categories. The irony is that many excellent programs are available from exactly these sources. Company policy should assess the pros and cons of imposing a blanket ban on its use. Again the establishment of software quality assurance to screen incoming disks is advised as it adds security and flexibility to software procurement.

It is also important to remember that shrink-wrapped software is not 'virus-free' by definition. (*See VB, May 1990, pp 4.*)

The best policy is:

- **restrict software to what is necessary for current projects**
- **the source of all software should be known definitively**

Dirty PC

Large organisations should establish one or more 'dirty PCs', which are used for running demonstration software, games, shareware and other programs not intended for company use. A dirty PC is an isolated machine and disks used on such a machine must **not** be installed on other computers. **Similarly no company software or data should be installed on a dirty PC.** It is important to isolate disks used on these machines and to keep an inventory of them. Redundant floppy disks can be wiped using the FORMAT A: command. Virus scanning software should be used regularly on dirty PCs.

The intention of providing these machines is to keep potentially virus-infected media away from company PCs. DP management can 'channel' viruses and other potentially dangerous software towards a single known source.

This concept is a powerful anti-virus tool, although it can be difficult to 'sell' to management if budgets or resources are strained. As PC prices are constantly falling, the relatively low cost of providing a dedicated machine (which need not be the latest model and can be second hand) should be treated as an insurance policy against virus attack.

Helpful Techniques

***** Evaluating software performance for general use is easier if the organisation adopts standard hardware and software configuration. Standardisation will also ease contingency planning and speed recovery in the event of a virus infection.

***** Access control should be installed and enforced. This limits the number of users and restricts their systems privileges. Access control is designed to impose *confidentiality* and is **not** recommended as a form of computer virus control. However, it will limit the number of people who can use the machine and install software.

***** Tell users to separate data disks from program disks and to ensure that program disks are write-protected.

***** If work is permitted on home computers, procedures should be devised to screen returned disks.

***** A system banner warning users about viruses and unauthorised software can be installed in AUTOEXEC.BAT. This message will appear every time the machine is booted.

Eight Golden Rules

1. Make regular and verified backups of your data.

2. Never boot from a diskette unless it is a clean system diskette and write-protected.

3. If you accidentally boot a machine from any floppy diskette, restart the machine from cold by switching off first.

4. Never leave floppy disks in the disk drive for longer than necessary.

5. Write protect all system diskettes containing system or program code (after making sure they are 'clean').

6. Don't copy program or system files from machine to machine, always copy from write-protected master diskettes.

7. Prevent or limit the exchange of program code over network and communications links.

8. Install suitable, reliable protection/detection software and use it.

Reproduced by permission of Bates Associates.

FEATURE ARTICLE

Fridrik Skulason

The Bulgarian Computer Viruses - 'The Virus Factory'

Just a few months ago most of the PC viruses known were thought to have been written in the United States or Western Europe. Only a single example (Dark Avenger) had been reported in Eastern Europe and few would have believed that over 70 virus variants would later originate from this region.

According to available sources, one of them, W13 was written in the Soviet Union and another, Vcomm, was written in Poland. The rest come from Bulgaria.

Why Bulgaria?

One possible contributory factor is that the availability of PCs is greater in Bulgaria than in the other countries in Eastern Europe.

Another possible explanation is that the viruses are politically motivated. Most PCs in Bulgaria are government property and creating viruses is a way of attacking the State. However, many of the Bulgarian viruses contain English text strings which suggests that they were intended to victimise PC users in other countries. American and UK popular culture accounts for some of these messages.

An important factor is the general attitude towards computer programming in the Communist bloc. Writing programs for profit and widespread use is not possible due to restrictions on private enterprise and the prevalence of software piracy.

The attitude of the Bulgarian programmers seems to have been "I am clever and I can write great programs, but if this does not make money then I will write a virus instead."

A competition seems to have developed between the Bulgarian virus writers, each trying to write the most compact virus, or the most technically proficient and ingenious example.

It has to be said that some of the Bulgarian viruses demonstrate a level of technical sophistication unkown elsewhere. Most of them have only been encountered in Bulgaria, although we are now receiving reports of infections in West Germany and ocasionally from other countries. **Travel restrictions have now been relaxed in Bulgaria and throughout the Eastern bloc. It is also no longer illegal to take computer programs out of these countries. These factors enormously increase the likelihood of infections occurring elsewhere.**

The Bulgarian Viruses in Chronological Order

1. *Vienna variants*	
2. *Old Yankee*	
3. *Vacsina/ Yankee*	
4. *Dark Avenger (Eddie)*	
5. *Pixel (Amstrad) variants*	***(VB, June '90)***
6. *Eddie 2*	
7. *Number of the Beast (V512, '666')*	***(VB, June '90)***
8. *Murphy*	***(VB, June '90)***
9. *800*	***(VB, June '90)***

Short descriptions and details for programmers were published in *VB, March 1990* unless otherwise indicated.

Vienna

As the name implies, the virus originated in Austria but soon found its way to Bulgaria. The modified version of Vienna which appeared in Burger's book *Computer Viruses: A High Tech Disease* also appeared in Bulgaria. Subsequent Bulgarian variants appeared which had been 'tidied up' to decrease program length. The author of the three shortest variants is known as T.P., but he is also responsible for the 'T.P. series' which contains the viruses 'Yankee' and 'Vacsina'. (*Jim Bates' analysis of the Vienna viruses will appear in next month's VB.*)

Old Yankee

This is believed to be the first Bulgarian virus which was written from scratch. It only infects .EXE files, increasing their length by 1961 bytes. When an infected program is run, the virus searches for an uninfected file. It uses a recursive depth-first search on the current drive, until all sub-directories have been searched or an uninfected file is found. This results in progressively longer delays when running an infected program. When a program has been infected, the virus plays the tune "Yankee Doodle Dandy" before passing control to the original program. It does not remain resident in memory. At the very end of the virus, the word 'motherfucker' appears. This is used as a signature to mark files as already infected.

Another version of this virus is shorter - only 1624 bytes long and it does not play the tune. This difference reduces the probability of detection, making the virus potentially more dangerous. **This variant was probably created from source code published by the original author. This irresponsible practice is becoming more and more common in Bulgaria.**

Vacsina/Yankee

This is a family of approximately fifty viruses written by a Bulgarian programmer called T.P. The viruses contain a version number system, which prevents an older variant from infecting a file already infected by a more recent version. A higher numbered version will automatically remove an older version before infecting a file - an automatic update sytem!

All the viruses are memory resident .EXE and .COM infectors. Their sizes range from 1200 bytes to 4000.

One of the first of these viruses found in the West (number 5) contained the string "VACSINA", which is Bulgarian for "vaccine". Apparently, T.P. had designed a device-driver based anti-virus program with the name of Vacsina and wanted the virus to remain inactive if the driver was found.

The viruses are "harmless" - that is they are not designed to cause damage. However, they produce various irritating side-effects and could be modified to contain detructive code.

The first four versions produce a short 'beep' when an infected program is run. Versions 5 - 25 have no noticeable side-effects. Version 26 includes code from the widely distributed Old Yankee virus. This version and all subsequent versions play "Yankee Doodle Dandy" under various differing circumstances. Versions 26 - 32 play it when Ctrl-Alt-Del is pressed, versions 33 - 43 play it when the system clock reaches 0500 or 1700, but versions 44 and above have only a 1-in-8 chance of playing the tune at these time of day.

Versions numbered below 37 infect .COM files in standard fashion, but .EXE files are infected in a highly unusual manner. All .EXE files contain 'MZ' or 'ZM' at the beginning. If these viruses find this marker, they overwrite it with a JMP to the end of the file exactly as if the file were .COM. However, the code appended to the end of the file is not the virus itself but rather a short 'loader' program which extracts the required information from the file header and relocates the file. This program is obviously based on one used in various versions of DOS in FORMAT.EXE and other programs. From a structural point of view, the file is now equivalent to a .COM file and will be infected as such the next time it is executed.

Versions 38 and above are radically different. They are functionally similar, but most of the virus code seems to have been rewritten. They infect .EXE files in the standard way.

Other changes were introduced as the programs evolved. Starting with version 33, the virus tries to foil any attempts to disassemble and patch it. This is achieved by intercepting INT 3H, as several other viruses do, and by using self-correcting *Hamming* code. These viruses can prohibit any modification of up to 16 bytes. Modifications are spotted and 'repaired'. Of course, the *Hamming* code itelf can be disabled allowing unlimited modification.

Versions 42 and above locate and destroy the ubiquitous Italian (Bouncing Ball, Ping-Pong) virus and version 46 also contains a routine to remove infections by Cascade (1701). The bizarre reasoning behind this seems to be that PC users are not to be trusted in detecting and removing existing viruses. Therefore, anti-virus programs must spread by their own means and execute without the intervention of the user, which is only possible if they are virus programs themselves!

Versions 50 and above are reported as being able to detect whether they are running on a 80286/80386 system, in which case they use protected mode instructions to circumvent memory resident interrupt monitoring software. This cannot yet be confirmed.

Eddie (Dark Avenger)

This virus is known as "Eddie' and its unknown author calls himself "Dark Avenger". A detailed description can be found in *VB, February 1990.* Eddie is probably the most widespread and dangerous Bulgarian virus to have appeared so far.**Since February, a new 2000 byte variant has appeared. More variants can be expected. The source code has been widely circulated.** Paradoxically, the author is understood to have distributed a disinfection program to remove known variants. This disinfection program includes a description of the development of this virus which is included here. Spelling and syntax errors are reproduced as in the original text file.

```
It may be of interest to you to know that Eddie
(also known as "Dark Avenger") is the most
widespread virus in Bulgaria for the time being.
However, I have information that Eddie is well
known in the USA, W. Germany and USSR too.

I started in writing the virus in early September
1988. In those times there were no any viruses in
Bulgaria, so I decided to write the first Bulgarian
virus. There were some different Eddie's versions:

VERSION 1.0, 31-OCT-1988
This version established the most important
features of the Eddie virus. Staying resident into
high end of memory, it was infecting .COM and .EXE
files, but only when executing them. INT 13 hadn't
been handled in any way. This version was damaging
infected files only, rather than infected disks.
Also, there weren't any messages in it (I still
wasn't choosed a name for it).

VERSION 1.1, 16-DEC-1988
In December I've decided to enhance the virus. This
version could infect files during their opening.
For that reason, a read buffer was allocated in
high end of memory, rather than using DOS function
48h when needed. The disk was destroyed instead of
the infected files.
```

```
VERSION 1.2, 19-DEC-1988
This added a new feature that causes (for example
compiled programs to be infected at once if the
virus is resident. Also, the "Eddie lives..."
message was added (can you guess why exactly
"Eddie?")

VERSION 1.31, 3-JAN-1989
This became the most common version of Eddie. A
code was added to find the INT 13 rom-vector on
many popular XT's and AT's. Also, other messages
were added so its length would be exactly 1800
bytes. There was a subsequent, 1.32 version (19-
JAN-1989), which added self-checksum and other
interesting features that was abandoned because it
was extremely buggy.

In early March 1989 Version 1.31 was called into
existence and started to live its own life to all
engineers' and other suckers' terror. And the last

VERSION 1.4, 17-OCT-1989
This was a bugfix for version 1.31, and added some
interesting features. Support has been added for
DOS 2.x and DOS 4.x. For further information about
this (the most terrible) version, and to learn how
to find out a program author by its code, or why
virus writers are still not dead, contact Mr.
Vesselin Bontchev (All rights reserved).

So, never say die! Eddie lives on and on and on...
Up the irons!
```

The paragraph about Vesselin Bontchev needs some explanation. Bontchev is an engineer and research associate at the *Institute of Industrial Cybernetics and Robotics* of the *Bulgarian Academy of Sciences*. He is the author of the most widely used anti-virus programs in Bulgaria and has provided much of the material that this article is based on, as well as samples of some of the Bulgarian viruses.

The 2000 byte variant (version 1.4), contains a check for programs written by Vesselin Bontchev. It scans every program before it is executed. If Bontchev's copyright message is found, the program will not execute and the infected computer freezes. **This is an early example of sabotage tactics to discredit anti-virus software authors.**

Two minor variants of version 1.4 have been found. One contains the numeric string '666' (presumably copied from The Number of the Beast virus) as well as:

```
Copy me - I want to travel
```

In the other variant, probably the original one, the following text is found:

```
Only the Good die young...
```

The virus writer, probably in his mid-twenties, is almost certainly a fan of the English heavy metal group Iron Maiden hence "up the irons". 'Eddie' is the band's skeleton mascot.

Pixel/Amstrad

The Pixel virus was written by a programmer called "Nico" in Greece and published in the computer magazine *Pixel*, in the form of a type-in BASIC program that would create an infected .COM file when run.

The original virus appended itself in front of infected programs increasing their length by 847 bytes. No effects are visible until the fifth generation is reached, when the virus has a 50 % chance of producing the following message when an infected program is run.

```
Program sick error: Call doctor or buy PIXEL for
cure description
```

An anti-virus program specific to this virus was subsequently published by *Pixel* magazine.

The virus does not remain resident in memory, but when it is activated all .COM files in the current directory will be infected.

A variant of this virus, known as "Amstrad" has also appeared. The only significant difference is a replacement of the above text with an advertisement for Amstrad computers. This variant is believed to have originated from Spain or Portugal.

The Bulgarian "improvements" to Pixel consisted of making it shorter, while preserving functionality. Three variants are known, one is 345 bytes long and another is only 299 bytes. The third is a miniscule 277 bytes in length, making it the smallest PC virus known. This third variant displays a message 'PARITY ERROR' when an infected program is run.

Eddie-II

Compared with some of the Bulgarian viruses, this one is fairly harmless, as it has no effects other than replication. It is called Eddie II because it contains the string "Eddie lives" which presumably refers to the original "Eddie" virus. Eddie II infects .EXE files as well as .COM files, but unlike most other .EXE infectors, it does not pad them to make their length a multiple of 16 bytes prior to infection. Infected files are marked with a value of 62 in the 'seconds' field of the timestamp, but as this method is also used by the Vienna and Zero Bug viruses, this makes programs infected with either of these viruses immune to infection by the others.

Infected files grow by 651 bytes, but this increase will not be seen if a 'DIR' command is given because the virus stays resident in memory, intercepting the 'find-first' and 'find-next' functions. If either function is called and the file found contains 62 in the 'seconds' field, the virus will decrement the file length by 651. If the file is a short program, infected by one

of the smaller variants of Vienna making the total length less than 651, the resulting file length will be negative. Since a file cannot have a negative length, DOS displays the unsigned binary equivalent resulting in a value of approximately 4 Gigabytes.

Number of the Beast, (aka V512, '666')

This virus was described in the May edition of *VB*. It infects .COM files by overwriting the first 512 bytes of the file. This part of the file is stored in the free space at the end of the file.

Three new variants have now appeared. They contain minor changes and corrections to some of the errors in the original virus. Due to space limitations, some error checking has been left out instead.The new versions do not contain the numeric string '666' at the end of the file.

Murphy

The variants of the Murphy virus are unusual for several reasons. To start with, the authors are known. Their names are Lubomir Mateev Mateev and Iani Lubomirov Brankov and their (confirmed) telephone numbers and addresses are listed in the source code they distributed.

Murphy is clearly based on the Eddie (Dark Avenger) virus as it incorporates several elements from it including the method to obtain the original INT 13H address, and the way in which the virus installs itself in memory and infects files.

Murphy is a memory resident virus capable of infecting .EXE and .COM files. **Unlike other viruses, it not only supports DOS 4.0, but also uses it, infecting files when function 6CH (extended open) is called.**

If an infected program is run between 10 and 11 a.m., the speaker produces a 'click' every time a DOS function is called. The resulting "shuffling" sound gives an indication of the activity of the computer - one can hear the computer 'think'.

Like many other viruses, it infects COMMAND.COM by overwriting the part of it located in high memory. When the infected program terminates, COMMAND.COM will be reloaded and infected.

A message is found in the virus which does not appear to be displayed on screen.

```
Hello, I'm Murphy. Nice to meet you friend. I'm
written since Nov/Dec. Copywrite (c) 1889 by Lubo &
Ian, Sofia, USM Laboratory.
```

"USM Laboratory" does not exist but ''Lubo & Ian" are the authors of the virus.

Two variants are known. The first has an infective length of 1227 bytes. It will only infect .COM with a length between 127 bytes and 64226 bytes, but programs longer than 64003 bytes prior to infection will not run when executed because the total length of the program, including the stack space required, will be above 64K. This version is understood to have been stolen and distributed before its author completed it.

A second "official release" version is 1521 bytes in length and contains several minor changes. Every exact hour, the virus jumps to the ROM BASIC interpreter. In all likelihood, this will cause a simple reset on many clones that do not have BASIC in ROM. The message inside the virus is shorter:

```
It's me - Murphy. Copywrite (c) 1990 by Lubo & Ian,
Sofia, USM Laboratory.
```

The method used to check whether the virus is present in memory has also been changed. The first version uses INT 21H function 4B59, but the second uses function 4B4D.

800

The 800 virus infects only.COM files and increases their length by 800 bytes. Unlike most other viruses, it does not simply append the virus code to the beginning or the end of the programs it infects. The virus writes its own code to a random section of the file, places a 3-byte JMP at the beginning of the file and appends the original, overwritten code at the end.

When an infected program is run, the virus will create an 8K "hole" in memory and transfer itself there. Like "The Number of the Beast", it uses an undocumented function to obtain the original INT 13H address, but instead of intercepting INT 21H, the virus intercepts "network hook" in DOS, INT 2A, function 82.

The virus has a simple encryption algorithm, but at the time of writing, it has not been fully disassembled. **The virus appears harmful, however, as it wiped out the root directory on drive C: on a computer it was being tested on, bypassing all installed anti-virus programs.**

Acknowledgements *to Mr. Vesselin Bontchev of the Bulgarian Academy of Science for supplying documentation, notes and samples used in the preparation of this article.*

NOTE: The list of search patterns so far published in *Virus Bulletin* will identify all of the known Bulgarian viruses. Many of the viruses display commonality or are minor variants making the extraction of numerous patterns unnecessary.

Refer to the box on page 6 of this edition, *VB* March 1990 and the *Known IBM PC Virus Table (Updates)* on page 11 of this month's edition.

ANTI-VIRUS SOFTWARE FOR IBM PCs

The following is a list of anti-virus software for IBM PC XT/AT and clones. Manufacturers are listed in alphabetical order followed by the trademark name of their associated product(s). New anti-virus products appear on page 20.

Advanced Software Protection Inc., PO Box 81270, Pittsburgh, PA 15217, USA. Tel. USA 412 422 4134	**ASP/ PATMAT**
All Software, Frederiksvarksgade 96, 3400 Hilleroed, Denmark. Tel. Denmark (45) 2 740303	**ALLCURE**
Bates Associates, Treble Clef House, 64 Welford Rd, Wigston Magna, Leicester LE8 1SL, UK. Tel. UK (0)533 883490	**SCANNER/ VIS SOFTWARE**
ComNETco Inc. 29 Olcott Square, Bernardsville, NJ 07924, USA. Tel USA 201 953 0322	**VIRUSAFE**
COMSEC, 5 Jabotinsky St, Ramat Gan 52520, POB 36890, Tel Aviv, Israel. Tel. Israel (972) 3 751 8113	**V-ANALYST**
Computer Integrity Corporation, PO Box 17721, Boulder, CO 80308, USA.	**VACCINATE**
Digital Dispatch Inc., 55 Lakeland Shores, St Paul, MN 55043, USA. Tel. USA 612 436 1000	**DATA PHYSICIAN**
Director Technologies Inc., 906 University Place, Evanston, IL 60201, USA. Tel. USA 312 491 2334	**DISK DEFENDER**
EliaShim Microcomputers Ltd. P.O.Box 8691, Haifa 31086, Israel. Tel. Israel (972) 4 523601	**VIRUSAFE**
Foundationware Inc., 2135 Renrock, Cleveland, OH 44118, USA. Tel. USA 216 932 7717	**CERTUS**
Gilmore Systems, PO Box 3831, Beverly Hills, CA 90212-0831, USA.	**X-FI-CHECK**
Gliss & Herweg GmbH, Post Box 2157, D-5024 Pulheim 2, West Germany. Tel. W. Germany (49) 2234 82227	**PC-CHECKUP**
IBM (Regional distributors worldwide)	**VIRSCAN**
IDS (International Data Security), 37-41 Gower Street, London WC1E 6HH, UK. Tel. UK (0)71 631 0548	**VIRUS-PRO**
Integrity Technologies Inc, 1395 Main Street, Metuchen, NJ 08840, USA.	**VIRALARM 2000 PC**
Interpath Corporation, 4423 Theeney Street, Santa Clara, CA 95054, USA.	**C-4**
Iris Software & Computers, Hamavo 6, Givataim 53303, Israel. Tel. Israel (972) 3 5715319	**ANTI-VIRUS**
Frisk Software, PO Box 7180, 127 Reykjavik, Iceland. Tel. Iceland (354) 1 17273	**F-PROT**
McAfee Associates, CVIA, 4423 Cheeney St, Santa Clara, CA 95054, USA. Tel USA 408 727 4559	**SCAN**
Microcom Software Division, PO Box 51816, Durham, NC 27717, USA, Tel. USA 617 551 1277	**VIREX-PC**
Microcraft Inc, PO Box 1652, Richmond, IN 47374, USA.	**VIR-X**
MSS Technology, The Graftons, Stamford New Rd, Altrincham, Cheshire, UK. Tel. (0)619 416429	**TERMINATOR**
Panda Systems, 801 Wilson Road, Wilmington, DE 19803, USA. Tel. USA 302 764 4722	**DR PANDA UTILITIES**
Paul Mace Software, 400 Williamson Way, Ashland, OR 97520, USA. Tel. USA 503 488 0224	**MACE VACCINE**
PC Security, The Old Court House, Trinity Road, Marlow SL7 3AN, UK. Tel. (0)628 890390	**ELIMINATOR**

Prime Factors Inc, 1470 East 20th Avenue, Eugene, OR 97403, USA. Tel. USA 503 345 4334	**VI-RAID**
Quaid Software Ltd. 45 Charles Street East, 3rd Floor, Toronto, Ontario, Canada M4Y 1S2. Tel. Canada 416 961 8243	**ANTIDOTE**
RG Software Systems, 2300 Computer Avenue, Suite A-7, Willow Grove, PA 19090, USA. Tel. USA 215 659 5300	**VI-SPY, DISK WATCHER**
S&S International, Weylands Court, Water Meadow, Germain Street, Chesham HP5 1LP, UK. Tel. UK (0)494 791900	**DR SOLOMON'S TOOLKIT**
SA Software, 28 Denbigh Road, London W13 8NH, UK. Tel. UK (0)71 998 2351	**PC IMMUNISE II**
Software Concepts Design, 594 Third Avenue, New York, NY 10016, USA. Tel. USA 212 889 6431	**FLUSHOT +**
Software Services, Niederweisstrasse 8, CH-5417 Untersiggenthal, Switzerland. Tel. Switzerland (41) 56 281116	**VIP**
Sophco Inc, PO Box 7430, Boulder, CO 80306, USA. Tel. USA 303 444 1542	**VACCINATE**
Sophos Ltd, 21 The Quadrant, Abingdon Science Park, Abingdon, Oxon OX 14 3YS, UK. Tel. (0)235 559933	**VACCINE/ SWEEP**
WorldWide Software Inc, 40 Exchange Place, 15th Floor, New York, NY 10005, USA. Tel. USA 212 422 4100	**VACCINE**
Zortech Inc, 366 Massachussetts Avenue, Arlington, MA 02174, USA. Tel. USA 617 646 6703	**CHECK-IT!**

KNOWN IBM PC VIRUSES (UPDATE)

Amendments and updates to the *Virus Bulletin Table of Known IBM PC Viruses*,28 May 1990. Hexadecimal patterns can be used to detect the presence of the virus using The Norton Utilities or your favourite scanning program.

EXAMINED VIRUSES

FISH 6 - CER: 3548 bytes. Partially based on 4K (4096). The virus is stored in encrypted form and the decryption routine is so short that a 16 bytes identification pattern is not possible.

```
FISH 6  E800 005B 81EB A90D B958 0D2E 8037 ; Offset 0
```

Liberty - CER: 2873 bytes. Virus comes from Indonesia.

```
Liberty 0174 031F 595B 5053 5152 1E06 1E0E 1FE8 ; Offset 080
```

Murphy - CER: Two variants exist, which produce a click from the loudspeaker when any DOS function is called.

```
Murphy B44A CD21 8CC0 488E D8C7 0601 0008 00E8 ; Offset variable
```

Number of The Beast - CR: Three new variants exist containing minor corrections to errors in the original virus.

```
Number of Beast (1) B800 3DCD 2193 5A52 0E1F 1E07 B102 B43F ; Offset variable
```

800 - CR: Awaiting disassembly.

```
800 Virus B981 0151 AD33 D0E2 FB59 3115 4747 E2FA ; Offset 00E
```

Korea - DR: Awaiting disassembly

```
Korea 31C0 8ED8 8ED0 BCF0 FFFB BB13 048B 0748 ; Offset 008
```

PrintScreen - DR: Occasionally performs a Print Screen (PrtSc) operation.

```
Printscreen FA33 C08E D0BC 00F0 1E16 1FA1 1304 2D02 ; Offset 023
```

VP - CN: Contains a variable number (1 to 15) of NOPs at the beginning followed by 909 bytes of virus code.

```
VP 0001 FCBF 0001 B910 00F2 A4B8 0001 FFE0 ; Offset variable
```

Pixel - CN: A third and new variant which is currently the shortest virus in existence. Displays the message "PARITY ERROR"

```
Pixel(3) 0001 0001 2E8C 1E02 018B C32E FF2E 0001 ; 277 bytes
```

REPORTED ONLY:- **5120** **Shake** - CER ? **Form** - DR

TOOLS & TECHNIQUES

Dr. Keith Jackson

Dynamic Decompression, LZEXE and the Virus Problem

Data compression is a means of reducing file size, for instance during transmission between modems. Five different compression programs are available for PCs, known as PKARC, PKZIP, LHARC, PAK and ZOO. These are freely available as shareware, and most facilitate the creation of compressed 'self extracting archive' files - executable programs containing the compressed content of one or more files, which on execution recreate disk copies of the original uncompressed files. Self extracting archive methods are often used to distribute programs, as they do not require a user to possess a copy of the appropriate decompression program, yet the size advantages of compression are still available. Compression of 50 % is not unusual, and text files can often be compressed even further to about 25% of their original size.

The time required to decompress files can be made much less than the compression time and, taken to its extreme, **decompression can happen dynamically at the time of program execution.** A program called LZEXE is now available which can perform dynamic decompression of a previously compressed file. This means that a program can be stored on disk in compressed form, loaded into memory, decompressed and then executed. LZEXE only works with executable files which have an extension of .EXE, and for such files the decompression time is barely noticeable. For example, I used LZEXE to compress an executable file which occupied 127Kbytes. This was reduced to an executable file of 67Kbytes (approximately 50% of the original size) with no effect whatsoever on program functionality, at the expense of increasing the load time by less than half a second (on a 386 PC). Using LZEXE to compress rarely used programs on my hard disk retrieved over 2 Mbytes of disk space; a significant saving.

During decompression LZEXE validates a checksum attached during compression, locates itself in high memory, decompresses the original file, adjusts the segment registers as necessary, and passes control to the decompressed program.

LZEXE is in the public domain so it can be freely copied and distributed. The documentation states that LZEXE can be used on any commercial software, therefore software authors may be tempted to use it to reduce the size of executable files. Many major companies already use one of the five data compression methods mentioned above to compress software for distribution (e.g. Borland International). This saves money for software distributors (programs do not require as many disks), and users save disk space. Everybody seems to gain.

Associated Dangers

Unfortunately there could be major problems with using programs such as LZEXE indiscriminately. **Decompression happens immediately before execution commences, therefore there is no chance to inspect the decompressed program before execution commences.** How can a user test that a program has not been infected by a virus **before** compression? Put succinctly, such tests seem to be difficult, if not impossible, to carry out. The standard method of searching for virus infection is to use one of the many programs available which search for specific hexadecimal patterns. The best scanning programs are regularly updated to keep track of new viruses as and when they are discovered. However, *any virus attached to an executable program (a parasitic virus) would be compressed in the same way as the executable program itself, rendering pattern searching useless* . A program compressed by LZEXE would make reliable detection by virus scanning programs impossible prior to execution.

How does a user tell whether an executable file has been compressed? The only practical way seems to be to search for a pattern inserted by the compression program. In the case of LZEXE, the author's name (Fabrice BELLARD) is found within the compressed executable file, but searching for this pattern will be insufficient when other dynamic decompression programs appear, as they surely will.

This presents a new and more sinister way that malicious programs of any type can be transmitted. It would be nice to think that LZEXE would be shunned when just one virus infected program compressed with this technique was discovered, but this is unlikely. Technically inept people who have obtained a virus may well be tempted to try out this simple method of virus distribution.

The documentation provided with LZEXE claims that compressed programs are less vulnerable to virus attack, as any alteration will cause the checksum inspected during decompression to fail. This is true for files that have already been compressed before virus infection takes place, but it misses the point. **Virus infection before compression is applied (deliberate or otherwise) is the real worry.**

I would not try and curtail the use of LZEXE, to attempt to do so would be to fly in the face of what is a very useful technique, but **distribution of programs compressed by LZEXE should be avoided.** They could contain any type of malicious program, and consequent detrimental effects cannot be prevented, they can only be detected after the event, which may be too late.

Technical Editor's Note: Some recent anti-virus programs scan LZEXE-packed files. If support for other compression programs is added, the main dangers posed by packing infected files will decrease.

WORM PROGRAMS

Last month, sentencing of Robert Morris, author of the notorious Internet worm, resulted in three years probation, a $10,000 fine and 400 hours of community work.

Morris is the first person convicted under the US Computer Fraud & Abuse Act 1986 which was designed to protect 'Federal interest' computers. However, computer security specialists are generally disappointed by the lenient sentence; Don Parker of SRI summed up the reaction: ''It's terrible. It's exactly what 10,000 hackers out there were hoping to hear.''

In this article, David Ferbrache provides a retrosprective on the most sensationalised and misreported 'virus' incident so far.

The Internet Worm - Action and Reaction

David Ferbrache

The *Defense Advanced Research Projects Agency* (DARPA) Internet is one of the world's principal academic and military research networks linking an estimated 60 thousand host systems sub-divided into 500 networks. This network was the subject of an attack by a worm program on Wednesday November 2nd 1988. The program, released by Robert T. Morris of *Cornell University*, was technically a virus which replicated under its own control infecting a variety of UNIX based host systems using the Internet protocol stack.

Genesis

The *Cornell University* mailer log files indicated that probing of an electronic mail handler (*sendmail*) was occuring from October 19th to October 28th. The *sendmail* program implements the Internet *simple mail transfer protocol* (SMTP). This program provides a wide range of mail handling and transfer functions, including the ability to invoke a program to process an incoming mail request. Such programs normally include "vacation" auto-mail reply and personal mail sorter utilities.

Sendmail - The Debug Option

Gene Spafford (of Purdue) has suggested that initial tests saught to transfer binary files directly using SMTP (a protocol which operates only on ASCII 7 bit data). To do so the author made use of the debug mode on the *sendmail* program which allowed him to route mail to a specified program. In this case the program was a specially crafted editor which removed the mail headers and submitted the body of the mail message to the UNIX shell for execution.

Such an attempt would fail due to control characters in the worm binary being interpreted by the SMTP mailer.

By Wednesday 2nd November the author had developed and was testing a second method using a short "C" vector program. This method, which was only too successful, worked by composing a mail message consisting of the source form of the vector program, together with commands to cause its compilation and execution. The vector program in turn organised the transfer of the actual worm binary.

```
debug                                  Switch on debug mode
mail from: </dev/null>                 No source address
rcpt to: <"|sed -e '1,/^$/'d| Invoke a command to
                                       remove the header
/bin/sh ; exit 0">                     and send the rest to
                                       the shell
data                                   Commands follow
...
```

Sendmail Command Sequence

Vector - A Carrier Program

The vector program executed on the remote host. This program connected to the worm on the infecting host and sent a challenge string. Following a successful challenge the vector transferred three binary files from the parent worm, these files contained: *A Sun 3 binary of the worm; a VAX binary of the worm;* and *the source code for the vector program itself.*

The vector program then overlayed itself with the shell command interpreter, leaving the shell connected to the parent worm. The parent worm then sent shell commands to attempt linking and execution of the Sun binary. If this failed, linking and execution of the VAX binary would be attempted. Finally all incriminating files would be removed.

Worm - An Overview

The newly executing worm employed a variety of camouflage techniques including:

1. Forking on a regular basis to change process ID and reset CPU process times

2. Zeroing its argument vector (argv)

3. Unlinking the worm binary file and killing its parent shell

4. Encrypting (trivially) all strings in the worm binary by XORing with hex 0X80

The worm operated by probing each network interface (returned by the netstat) command. The worm built a list of all connected networks, including destination hosts (point-point links), gateway hosts and alternate host names. The host equivalence file was also used as a source of host-names.

The list of hosts was randomised by the worm to avoid duplication of effort by other worm instances. Possible host numbers on each network were generated. The host Internet addresses were then probed using the *telnet* or *rexec* port. This probe established that the host did exist, and in the case of rexec that it was a BSD UNIX system running the *Berkeley r protocols.*

Infection Routes

Three routes of infection were attempted, the sendmail attack has been previously addressed, the others are:

rsh - a **remote shell service** operating on distributed trust, by which a user can allow logins from trusted hosts without password. If successful the vector program was transferred, compiled and executed.

fingerd - **the finger service** (a utility to allow determination of the name and status of a user by username), had a serious bug which allowed the worm to cause the program to overlay itself with a copy of the shell. This shell again inherited the open sockets of its parent (connected to the worm). This was achieved through the flawed use of "gets" by the *fingerd* program. The code only attacked VAX systems in this manner.

The use of the "C" library call "gets" has been identified as a major problem in many utilities. This routine reads a string of characters from a file (or communications stream) but does not verify that the string will not overflow the target buffer. This overflow may be selectively manipulated (as in the Internet worm) to cause the machine stack to be modified, thus changing the flow of control in the program. It has been suggested that users are forced to use the alternative "fgets" routine by removing "gets" from the C library.

Finally, the worm entered its core finite state machine. This consisted of four stages each run for a short interval. After this the worm would use *sendmail*, *rsh* and *fingerd* to infect hosts within its database, before restarting the cycle.

1. Read the /etc/hosts.equiv, /.rhosts and forward files in each home directory to determine closely linked hosts

2. Attack each user password using no password, the account name and the GECOS (full name) field.

3. Attack each user password by using an internal dictionary of 432 words

4. Attack each user password by trying each word in the online UNIX dictionary

When a password was broken using one of the above attacks the worm would:

1. Try to invoke *rexec* on the hosts given in the *users.forward* and *.rhosts* file with the broken password. Since users often share passwords over networks this was a potentially succesful attack.

2. Try to use *rexec* on the local host to change to the user's identity followed by a *rsh* to the remote host. This relied on the local host appearing in the remote hosts *.rhosts* or *hosts.equiv* file.

Infection - How Successful Was The Worm?

All times in the following discussion have been amended to GMT, and are given in the form date/time. Testing of the revised worm occured at 02/2001. The final worm was released at 03/0100 from a host at *MIT* (edu.mit.ai.prep) via remote login from *Cornell.* The following table gives an example of the spread of the worm:

Infection reports

03/0200	University of Stanford
03/0224	Rand Corporation in Santa Monica
03/0230	MIT workstation pool
03/0304	University of California at Berkeley
03/0354	University of Maryland
03/0400	MIT AI labs infected
03/0449	University of Utah
03/0500	Stanford Research Institute
03/0700	Lawrence Livermore National Laboratory
03/0900	Project Athena workstation

Current estimates vary from 1-3,000 hosts infected. The worm was targeted against only Sun-3 and Dec Vax systems running UNIX. This machine base was a small subset of the hosts connected to the Internet. It is however significant that the worm included code to allow up to 19 object modules of which only two were used. The diversity of Internet host platforms was a significant factor in reducing the spread of the worm. Had Morris extended his attacks to the common Sun 386i and Sun 4 systems the impact would have been far greater.

First Sight

The first warning of the worm was posted by Peter Yee of NASA Ames on 03/0728 and stated that:

```
We are currently under attack from an Internet
VIRUS. It has hit UC Berkeley, UC San Diego,
Lawrence Livermore, Standford, and NASA Ames. The
virus comes in via SMTP and is then able to attack
all 4.3 BSD and SUN (3.X?) machines...
```

This warning was sent to the *tcp-ip mailing list*. This mailing list, although described as an obscure electronic bulletin board by the *New York Times*, was read by most administrators and developers of the Internet protocol suite.

Berkeley Reacts

Keith Bostic of the University of California at Berkeley (developers of the BSD Unix kernel whose utilities were attacked), reacted rapidly to distribute a number of fixes including a patch to disable the *sendmail* debug command, a suggestion to rename the "C" compiler and linker and a modified source for the *fingerd* program (which contained a bug exploited by the worm).

While these fixes received a widespread distribution (the initial fix being posted at 03/1058 to a wide range of newsgroups and mailing lists), it is worrying to note the delay with which vendors issued patches incorporating this advice.

In addition to the Berkeley fixes a number of independent fixes were produced including the so called "condom". This consisted of adding a single file to the */usr/tmp* directory to block the propagation of the vector program.

The repository of BSD UNIX fixes maintained by *Berkeley*, together with the *comp.bugs.4bsd.ucb-fixes newgroup* form a vital resource for system administators attempting to patch known security holes.

Administrative Practices

The Internet is a diverse network linking numerous hosts and organisations. No requirements concerning security, integrity or administrative practices are made before allowing a host to be connected to the Internet (other than observance of the network protocol specifications). Thus the level of expertise of the systems administrators varies greatly, as does their willingness to apply vendor patches or public software fixes. It must be stated that the Internet functioned perfectly, and that it was the security of the host application layer that was flawed. The Internet applications layer comprises *telnet, smtp, the Berkeley r protocols (rcp, rsh, rexec), finger* and many other services. It is vital that the code of such applications is closely scrutinised.

Network layer	**Transport layer**	**Application layer**
Internet protocol	Transmission control	*Telnet* *Smtp* *Rexec* *Finger* *Rsh*

Internet Protocol Stack

Old Boy Network

The principal work on disassembly and analysis of the Internet worm was done through the co-operation of a number of researchers in the computer security and communications fields through the 'old boy network'. This system, although difficult to formalise, gave a great deal of flexibility and allowed the pooling of much experience. There is, however, little doubt that much initial work was duplicated, with people working through the night using basic tools such as *adb* on captured coredumps. The construction of more advanced disassembly and automated decompilation tools has been identified as important to minimise response time.

Phage-l

At 04/0220 Gene Spafford established the cornerstone of joint effort into the worm, this was the *phage* mailing list. Initially approximately 33 addresses appeared on the list representing companies, network relay sites and academic institutions. The issue of trust both in email and personal communication represented a major problem.

How could a researcher authenticate himself in any communication? Personal knowledge remained the principal route for telephone communication. The originating address of email served as authentication, although it is comparatively easy to forge *smtp* mail addresses.

This issue, together with the issue of email security, has now been addressed by the Internet community who have published a number of draft request for comments discussing authentication and encryption methods for email exchange.

Connectivity

The issue of network connectivity was a dual edged weapon. While the rapid analysis and posting of bug fixes and information clearly enabled a number of hosts to avoid or deal with infection, the widespread closely coupled network topology hastened the spread of the worm. As standards emerge for the Internet protocols, open systems protocols (ISO, OSI) and for the UNIX operating system, a repetition of the worm will become very likely.

There is a clear need for a reserve distinct communication channel through which rapid reporting of security problems and network alerts can be propagated. Where this is not possible, it seems certain that the facility to route priority messages (which bypass mail queues) is vital.

The disconnection of the Internet-MILNET mail bridges from 03/1630 - 04/1600 was an example of a mixed blessing. It prevented new occurences of the worm from crossing, but also effectively prevented the distribution of many of the bug fixes and reports.

Congestion on the Internet also prevented one critical message from Andy Sudduth from reaching the world. This message sent anonymously at the request of Morris, contained a warning of the worm and detailed briefly the attack methods used by the worm. In the event the message was queued at a major gateway site (*net.cs.relay*) for 2 days before being forwarded to the *tcp-ip list*. The message stated:

```
A possible virus report:
There may be a virus loose on the internet.
Here is the gist of a message I got:
I'm sorry.
Here are some steps to prevent further transmission:

1) don't run fingerd, or fix it to not overrun its
stack when reading arguments.

2) recompile sendmail w/o DEBUG defined

3) don't run rexecd

Hope this helps, but more, I hope it is a hoax.
```

Source Code Policy

The workers in the field developed a consensus not to publish or distribute decompiled virus code. This decision was made to reduce the possibility of a simple modification being made to the worm code to incorporate a new attack mechanism, followed by re-release of the worm. The code for the DECNET xmas worm and BITNET rexx chain letter(*see VB, April '90*) have been widely distributed, and it is worth noting that both were re-released on their first anniversary.

The source code for the worm was finally recovered from *Cornell University* backup tapes of Morris's account. This source, dated October 15th and November 2nd, encrypted by the insecure UNIX crypt command (based on the German *enigma* rotor machine), was decrypted by the computer centre at *Cornell*. It is functionally identical to the decompiled worm source.

The UNIX crypt command is less secure than the newer DES or public key algorithms, in fact a program, *Crypt Breaker's Workshop*, exists in the public domain to break such ciphers.

Media Reaction

On Thursday evening the media arrived. Media reports often exaggerated the damage, made great play on the connection of military systems to the Internet and utilised the terminology of virus and worm interchangeably. One report even showed IBM PCs in the background when discussing the worm, leading to great confusion. The use of a press officer to deal with media queries became vital to avoid disturbance of detailed analysis work under way. A typical press conference at MIT on Friday morning was attended by 10 TV crews and 25 reporters.

New York Times, Friday 4th November.

Virus in Military Computers Disrupts Systems

Nationwide Experts call it the largest assault ever on the nation's systems

New York Times, Saturday 5th November.

Author of Computer 'Virus' Is Son Of US Electronic Security Expert

Cornell Graduate Student Described as 'Brilliant'

Letter Bomb of the Computer Age

Virus Eliminated Defense Aids Say

Key Networks Are Said To Be Impossible To Penetrate

On the Front Lines in Battling Electronic Invader

New York Times, Tuesday 8th November.

Learning to Love the Computer Whiz

A winning team: 'slightly crazy people working late at night'

Organisation And Response

In the aftermath of the Internet worm the *Computer Emergency Response Team* was established. This group at the *Software Engineering Institute* at *Carnegie Mellon University* was formed to combat any future incidents like the Internet worm. DARPA has defined CERT's main functions as being the provision of:

mechanisms for coordinating community response in emergency situations, such as virus attacks or rumors of attacks;

a coordination point for dealing with information about vulnerabilities and fixes;

a focal point for discussion of proactive security measures, coordination, and security awareness among Internet users.

To date the CERT has produced a series of advisory messages:

CA8801	Ftpd vulnerability	*(v5.59 sendmail and earlier)*
CA8902	Sun restore hole	*4.0, 4.0.1, 4.0.3 SunOS*
CA8903	Telnet breakin warning	
CA8904	WANK worm on SPAN network	
CA8906	DEC/Ultrix 3.0 systems	
CA8907	Sun RCP vulnerability	*4.0, 4.0.1, 4.0.3 SunOS*
CA9001	Sun Sendmail vulnerable	*Up to 4.0.3 SunOS*

The CERT can be contacted by email at cert@edu.cmu.sei.cert, or telephone USA 412 268 7090 (24 hour hotline). CERT also maintains a mailing list for tools which can be useful in combating security incidents or improving system security, this mailing list can be joined by sending mail to cert-tools-request@edu.cmu.sei.cert.

Similar organisations have been established by the US *Department of Energy (Computer Incident Advisory Capability)*, the US *Defence Data Network* (security co-ordination center) and by *NASA* (SPAN network centre). Each co-operates closely in exchanging information on known security problems.

The *Virus-L* mailing list provides an additional informal network for reporting of network worms and other problems. An emergency (unmoderated) list address has been set up at valert-l@ edu.lehigh.cc.ibm1 which will send a warning to all recipients of the list worldwide.

Finally, the *Zardoz* closed UNIX security mailing list has established an emergency alert address at security-emergency@com.cpd.uninet. This mail address will mail to all users and administrators with a legitimate interest in UNIX security.

In the UK no comparable organisation exists to date despite attempts by the *Computer Threat Research Association* (CoTRA) to lay such a groundwork. It is hoped that the need for such an organisation (with international links) is recognised, and that a publicised counterpart to CERT established.

Policy, Ethics And Law

The worm also prompted the development of a series of policy statements by the network co-ordinators, an example is the Internet Activity Board (the regulatory body for the Internet) policy statement which states that:

Any activity which

1. seeks to gain unauthorised access to the resources of the Internet,

2. disrupts the intended use of the Internet,

3. wastes resources (people, capacity, computer) through such actions,

4. destroys the integrity of computer based information and/or

5. comprimises the privacy of users

is unethical and unacceptable. The board has backed this statement by an incentive to develop Internet security (under the auspices of the *Internet Engineering Task Force , working group 8*), and to improve the integrity of the Internet. **It should be noted that the Internet is currently highly vulnerable to denial of service attacks.**

This policy statement has the backing of the Californian penal code, and in the case of abuse of federal systems, the US *Computer Fraud and Abuse Act* of 1986 (18 USC 1030). Morris was indicted under this act which made it a crime to:

Intentionally, without authorisation, access a federal computer, or a federally used computer if such access affects the government's operation of the computer.

Summary Of The Incident

The *General Accounting Office* discussed the vulnerabilities exposed by the worm under the heading of:

1. Lack of a focal point to address Internet-wide security problems

2. Host weaknesses facilitated spread of the virus

3. Inadequate attention to security

4. System managers who are technically weak

5. Problems in developing, distributing and installing software fixes

6. Problems with vendors

Co-ordination and competent administration are key needs underlined by the worm incident.

References

US *General Accounting Office*, report to the chairman, subcommittee on telecommunications and finance, committee on energy and commerce. Computer Security: Virus Highlights Need For Improved Internet Management.

The Computer Worm. A report to the provost of *Cornell University* on an investigation conducted by the commmission of preliminary enquiry.

With Microscope and Tweezers; An Analysis of the Internet Virus of Nov 1988, M.Eichin, J.Rochlis, Massachusetts Institute of Technology.

A Tour of the Worm, D. Seeley, Dept of Computer Science, *University of Utah.*

The Internet Worm Program: an Analysis, Purdue technical report CSD-TR-823, E.Spafford.

Communications of the ACM, Volume 32, No 6, special section on the Internet worm.

Ethics and the Internet, Request for Comments 1087, *Internet Activity Board*

The Internet Activity Board, Request for Comments 1120, V.Cerf.

PRODUCT REVIEW

Dr. Keith Jackson

*This a preliminary assessment of FoundationWare's Certus package and its "Tools to Fight Viruses With". A further evaluation will be conducted on a 'standard' PC configuration as **dictated** by the software. FoundationWare are currently shipping Certus Version 2.00*

Certus

The Certus package claims to provide tools for "software security, quality assurance, usage control, auditing and hard disk recovery"

The copy of Certus provided for evaluation came on 5.25 inch (360K) floppy disks, and although the manual states that a hard disk is "recommended", in reality the manual discusses little else, and many of the features made available by Certus don't make much sense unless a hard disk is used as approximately 1Mbyte of disk storage space is required. This fact became rather crucial as I looked further into Certus (*see below*).

The Manual

The manual that accompanies Certus is 162 pages long, A5 size, in the standard three ring binder used by most software packages. It contains a thorough table of contents (seven pages long), a 6 page glossary, but no index. The lack of an index is inexcusable in a manual of this size and complexity. The Certus manual appears to be thorough, if somewhat repetitive.

The manual warns in strident terms against aborting the Certus installation process until it has completed its tasks. However it fails to explain why this is so important, and what could happen if installation ever were aborted (accidentally or otherwise). Although installation of Certus manipulates the hard disk at a low level, I could not find anything in the manual which explained that **taking a complete backup (or two) of the hard disk before installing such a package is a good idea.** There are other curious omissions.

The manual uses the phrase "Quality Assurance Level", with regard to Certus testing, and explains that it runs on a scale of 0 to 9. However there is no explanation of what this scale actually represents. I would hazard a guess that when checksumming tests are carried out, it refers to the interval between the bytes that are to be included in the checksum process, thereby trading off speed of checking versus security. However this is only a guess, and such matters need to be explained.

'SHELTER', 'QUICK' and 'The Blue Disk'

Certus includes many utility programs (tools). One of these is SHELTER, which creates a 'Critical Disk' which can be used to recover the File Allocation Table (FAT) of the hard disk if it ever proves necessary, and allows 'locking' of the hard disk. The Critical Disk contain copies of the hard disk's partition table, boot track, FAT, root directory and the information stored within CMOS RAM. It therefore contains all the necessary information to restore a disk but, as the manual states quite clearly, it must be appreciated that a Critical Disk only applies to one specific computer. Using it on another computer can result in chaos. The manual states that this Critical Disk allows "complete and automatic recovery from hard disk crashes". Short of crashing a hard disk deliberately, it is difficult to test such claims.

The utility program QUICK adds signatures to a database of files that are to be tested by means of a checksumming process. It is meant to be used by a system manger when he is configuring a computer. Coupled with the fairly standard facility of being able to specify which files should be checked by Certus, is a facility that I have not come across before, known colloquially as the Certus "Blue Disk". Such a disk is provided with each copy of Certus, updated frequently, and contains checksums that are known to be correct for most of the popular software packages. Even given the limitation that these checksums need to be updated whenever a new version of the application software is installed, I can well see how a large company with many computers could find such a service useful. It is also reassuring to know what the valid checksum should be, as checksum tests are useless if the file being tested has already been corrupted. The Blue Disk scheme precludes such problems.

Illegal disk writing (including prevention of hard disk formatting) is prevented by a utility called SURVEY, a small memory resident program which requires about 5K of RAM.

Passwords

The manual goes to great lengths to explain that passwords used by Certus are case sensitive. This is just plain silly. Although case sensitivity creates a slightly larger number of possible password combinations, it makes passwords very difficult to use. It is all too easy to set a password with the Caps Lock key inadvertently enabled, and then wonder why the password is always rejected when it is re-entered at some future date.

Secure Bootstrapping

The Certus manual describes two methods which purport to prevent a computer being booted from a floppy disk. I was intrigued by this claim, but all is not well with these methods.

The first method comprises altering the setup information held in CMOS memory so that the computer does not know that the first floppy disk drive exists. The manual warns that if you decide to do this, and the hard disk suffers a head crash, then the system "can be inaccessible". The only way to retrieve such a situation is to remove the battery which sustains the information in the CMOS RAM, and begin again as if you had a new computer.

The second method of preventing booting from a floppy drive simply comprises removing the ribbon cable from the rear of the disk drive. I find it difficult to take either of these methods seriously. They are cumbersome, could prevent usage of the floppy drive, and may interfere with maintaining proper backups. Altering the CMOS RAM does **not** work on a PC or an XT, such computers don't store setup information in CMOS RAM (as the manual acknowledges). I could go on, but suffice it to say that I recommend that neither method is used.

If you wish to prevent booting from the floppy drive, and wish to still have use of the floppy drive, then some hardware addition or alteration is required.

DOS Version Compatibility

Certus requires at least version 3.00 of MS-DOS (*see technical details below*). Anyone who still uses an earlier version of MS-DOS v2.11, cannot use Certus without upgrading the operating system. This is especially pertinent on portables such as the Toshiba T1000, where MS-DOS v2.11 is resident in ROM.

Machine Configuration

Now to install Certus; this was where my problems began.

I followed the installation instructions completely, even foregoing the temptation to use the chapter entitled "*Installation for those who refuse to read manuals*". However I found it impossible to install Certus on the computer which I usually use for testing purposes. As described in the technical details given below, this computer has three floppy disk drives, and one hard disk drive. The floppy disk drives are defined as drives A, B, and C, leaving the hard disk as drive D. MS-DOS is perfectly happy with this arrangement, and with a couple of honourable exceptions whose names will not be mentioned here, application software packages are also quite happy with this arrangement. Certus unfortunately was not happy, and insisted on trying to install on drive C.

I could find no way to prevent Certus insisting that drive C was the place that it should be installed. I tried to placate the installation program by placing a floppy disk in drive C. This produced the error message "*Error 32 has occurred. Insufficient disk space to install Certus*". This is hardly surprising given that Certus requires approximately 1MB of disk space. However it did point out another problem, none of the possible errors reported by Certus are documented anywhere in the manual (or even discussed).

I resorted to inspecting the content of the installation program INSTALL.EXE. Within this program are found pathnames C:\CERTUS\RESIDENT, C:\CERTUS\CERTUS.DAT, C:\CERTUS\CERTUS.OVL, and C:\CERTUS\HISTORY. There are many more such hard coded pathnames than the few quoted above, 74 more to be precise. If such pathnames are used by INSTALL to try and locate Certus files, then it is no wonder that INSTALL insists on accessing drive C. Words fail me when trying to describe such programming practices. If this is the cause of Certus insisting upon accessing drive C during installation (as seems probable), then even if I had four hard disk partitions as drives C, D, E and F, Certus would still insist on being installed on drive C, and in a fixed directory of \CERTUS.

Conclusion

What to say in conclusion? Certus may well be very good, and the features outlined above seem worth investigating. However, the software obstinately refused installation and I would warn you to be extremely careful with Certus if your computer does not conform exactly to the drive A=floppy disk, drive C=hard disk configuration found on most PCs. **Ultimately, it is for the user, not the software producer, to dictate the configuration of his or her PC albeit within the limitations of DOS.**

What did I like about Certus? The spelling mistake in the manual where computer memory is described as being made from 'silicone' (used for female breast enlargement among other things) rather than 'silicon' made me laugh a lot.

Technical Details

Product: Certus

Developer and Vendor: Foundationware, 13110 Shaker Square, Cleveland, Ohio 44120, U.S.A., Tel USA 216 752-8181, Fax USA 216 752-8188.

Availability: IBM PC,XT,AT,AT386,PS/2 and compatibles. A hard disk is recommended, and at least 384K of RAM is required, of which up to 8.5K can be used by memory resident software. Any version of MS-DOS from v3.0 upwards can be used.

Version Evaluated: 1.2f

Serial Number: 1-6206

Price: $189.00 (CERTUS LAN $795)

Hardware Used: ITT XTRA (a PC compatible) with a 4.77MHz 8088 processor, one 3.5 inch (720K) drive, two 5.25 inch (360K) drives, and a 30 Mbyte Western Digital Hardcard, running under MS-DOS v3.30.

END-NOTES & NEWS

A **bogus version of popular compression utility PKZIP** has been reported by Phil Katz, President of PKWARE. The current version of PKZIP is V1.10 and the bogus version purports to be V1.20. It is not known whether the bogus version has been Trojanised. A reward is offered to anyone providing information leading to the prosecution of the person(s) responsible for creating the bogus copy of PKZIP. Tel PKWARE, USA, 414 352 3670.

New and updated anti-virus products include:

PC-cillin "State of the art virus prevention". "Guaranteed to scan, detect and prevent at least 70 international viruses". *Trend Micro Devices Inc*., Tel USA 213 328 5892.

Virex-PC. IBM PC virus detection utility from *Microcom Software Division*, distributors of Virex Macintosh anti-virus program. Tel USA 617 551 1957 or 919 490 1277.

PC Immunise II. Improved virus non-specific package from *SA Software*. Earlier release was reviewed in *VB*, August 1989. Tel UK 071 998 2351.

Hyperaccess/5 PC communications package incorporating 'real time virus filtering technology'. Designed for screening downloaded software and compressed software. *Firefox Communications.* Tel UK 0784 430069.

Eliminator. Virus-specific monitoring and disinfection program. Updated quarterly.*PC Security Ltd.* Tel UK 0628 890390.

Reports include:

PC Viruses - "Reports From the Front Lines". A free report from Raymond Glath, developer of Vi-Spy (*VB*, May, 1990) and Disk Watcher. Discusses 'stealth viruses', software developments and the current extent of the problem. *RG Software Systems.* Tel USA 215 659 5300.

The Kinetics of Computer Virus Replication. *FoundationWare's* shock report predicting a global computer virus epidemic. New research which *VB* hopes to report shortly disputes the findings of this report. Tel USA 212 752 8181.

Computer Viruses, ''a definitive survival guide'' is a 262 page book from the *National Computer Security Association* and costs $55. Information from NCSA, Suite 309, 4401-A, Connecticut Av NW, Washington DC 2008, USA.

VIRUS BULLETIN

Subscription price for 1 year (12 issues) including delivery:

US$ for USA (first class airmail) $350, Rest of the World (first class airmail) £195

Editorial enquiries, subscription enquiries, orders and payments:

Virus Bulletin Ltd, 21 The Quadrant, Abingdon Science Park, Abingdon, OX14 3YS, England

Tel (0235) 555139, International Tel (+44) 235 555139
Fax (0235) 559935, International Fax (+44) 235 559935

US subscriptions only:

June Jordan, Virus Bulletin, 590 Danbury Road, Ridgefield, CT 06877, USA
Tel 203 431 8720, Fax 203 431 8165