VIRUS BULLETIN INTERNATIONAL CONFERENCE

Boston Park Plaza Hotel and Towers
20 - 22 September

95

000001

VIRUS BULLETIN

THE INTERNATIONAL PUBLICATION ON COMPUTER VIRUS PREVENTION, RECOGNITION AND REMOVAL

Welcome!

Welcome to *VB '95* in Boston - the *Fifth International Virus Bulletin Conference*. This is the first time the event has been held outside Europe and the interest from US delegates has been overwhelming.

A conference is always judged on the quality of its speakers and, this year, as always, I am delighted that the *VB* programme features so many of the world's authorities on viruses and IT security. I would like to thank all of the speakers for joining us here, especially Harold Highland and Chris Baxter, who have stood in at such short notice. I must also thank a number of people behind the scenes for their enthusiasm and hard work in organising the event; specifically, Karen Richardson, Julia Line, Dale Tabrum, Megan Palfrey, Ian Whalley, Jan Hruska, Beth Cece, Chris Perkins and his team, Alan Cummings, Bob O'Neil and Steve Holstein.

I hope that you will find the next two and a half days informative, thought-provoking and, most of all, fun. The conference programme is supported by various social functions which provide an ideal opportunity for informal discussion with speakers and delegates alike. On Wednesday, from 7.30pm - 8.30pm, there will be a 'welcome' drinks reception in the Plaza Ballroom, sponsored by *McAfee Associates*. And on Thursday evening, we will be holding a spectacular black tie gala dinner: pre-dinner drinks will be served from 7.30pm - 8.00pm in the Plaza Ballroom.

Please be kind enough to wear your badge at all times during the conference. Badge colours are: orange for delegates, cream for speakers, green for exhibitors, blue for press representatives and pink for the conference organisers. You will also be required to surrender the appropriate tickets for all meals. Should you have **any** problems during the conference, please do not hesitate to contact one of the organisers - identified not only by their badges but also by their very fetching *VB* shirts.

Enjoy the conference!

[illegible]

Petra Duffield
Conference Manager

P.S. We appreciate your comments (good or bad) on any aspect of the conference and would be grateful if you could complete and return the enclosed assessment form.

000002

THE CONFERENCE PROGRAMME

WEDNESDAY, SEPTEMBER 20

GEORGIAN ROOM		PLAZA BALLROOM		ARLINGTON ROOM
Jan Hruska, Sophos Plc, UK *An introduction to computer viruses*	16.00 to 18.00			

THURSDAY, SEPTEMBER 21

GEORGIAN ROOM		PLAZA BALLROOM		ARLINGTON ROOM
Ian Whalley, Virus Bulletin, UK *Opening address*	09.15			
Harold Highland, USA *Keynote address*	09.45			
	10.45	TEA / COFFEE	10.45	
Sarah Gordon, Command Software, USA *The anti-virus strategy system*	11.00	**Jonathan Lettvin**, Lotus Development Corporation, USA *The PC boot sequence, its risks and opportunities*	11.00	**Frans Veldman**, ESaSS GmbH, The Netherlands *Heuristics*
Paul Ducklin, Sophos Plc, UK *Blessings in disguise: building out of disaster*	11.45	**Neville Bulsara**, Quantum System Software, India *Securing DOS*	11.45	**Morton Swimmer**, Virus Test Center, Germany *Dynamic detection and classification of computer viruses*
	12.30	LUNCH	12.30	
Jean Hitchings, University of Nottingham, UK *Human dimension of computer viruses*	14.00	**Dmitry Mostovoy**, DialogueScience, Russia *Modern methods of detecting and eradicating viruses*	14.00	**Jakub Kaminski**, Cybec Pty, Australia *Flash BIOS - a new security loophole*
Mike Lambert, Frontier Corporation, USA *Fully automated response for in the wild viruses (FAR-ITW)*	14.45	**Shane Coursen**, Symantec Corporation, USA *Windows NT and Windows 95 vulnerability*	14.45	**Ferenc Leitold**, Hunix Ltd, Hungary *Automatic virus analyser system*
	15.30	TEA / COFFEE	15.30	
Wes Ames, Boeing Corporation, USA *Discussion forum: problems encountered by computer security managers of medium to large corporate organisations and how they are addressed by the developers of anti-virus products*	15.45	**Peter Radatti**, Cybersoft Inc, USA *Computer viruses in heterogeneous Unix networks*	15.45	**Igor Muttik**, Moscow State University, Russia *The problems in creating goat files*
	16.30	**Scott Gordon**, McAfee Associates, USA *Evaluating distributed virus protection products*	16.30	**David Aubrey-Jones**, Reflex Magnetics, UK *Automatic testing of memory resident anti-virus software*
	17.15	CLOSE OF DAY 1	17.15	

000003

FRIDAY, SEPTEMBER 22

GEORGIAN ROOM		PLAZA BALLROOM		ARLINGTON ROOM
Paul Robinson, Secure Computing, UK *A testing time*	10.00	**Robin Kinney**, Varian Oncology Systems, USA *Virus detection as part of the software development process*	10.00	**Righard Zwienenberg**, CSE, The Netherlands *Heuristic scanners: artificial intelligence?*
	10.45	TEA / COFFEE	10.45	
Chris Baxter, CCTA, UK *Government certification of anti-virus software*	11.00	**Igor Grebert**, McAfee Associates, USA *NetWare vulnerability*	11.00	**Glenn Coates**, Staffordshire University, UK *Virus detection - 'the brainy way'*
Judy Edwards, Illinois State University, USA *Fending off viruses in the university community*	11.45	**Pavel Lamacka**, HT Computers Ltd, Slovak Republic *Harmless and useful viruses can hardly exist*	11.45	**Roger Riordan**, Cybec Pty Ltd, Australia *Data security problems associated with IDE hard disks*
	12.30	LUNCH	12.30	
Allan Dyer, Yui Kee Company, Hong Kong *Recent viruses, writers and virus spread in Hong Kong and China*	14.00	**John Morar**, IBM Corporation, USA *The effects of computer viruses on OS/2 and Warp*	14.00	**Fridrik Skulason**, Frisk Software International, Iceland *Latest trends in polymorphism*
Lucijan Caric, Croatia *Case study of virus control in a large organisation*	14.45	**Richard Ford**, NCSA, USA *Boot sector virus removal under NT, OS/2, Windows 95, Unix and NetWare servers*	14.45	**Dmitry Gryaznov**, S&S International, UK *Scanners of the year 2000: heuristics*
	15.30	TEA / COFFEE	15.30	
Steve White, IBM Corporation, USA *Computer viruses: a global perspective*	15.45	**Jim Bates**, Computer Forensics Ltd, UK *Catching the virus writers*	15.45	**David Stang**, Norman Data Defense Systems Inc, USA *Computer viruses and artificial intelligence*
Speakers' Panel Session	16.30		16.30	
	17.30	CLOSE OF CONFERENCE	17.30	

EXHIBITION OPENING TIMES

STANBORO ROOM

Wednesday, September 20	14.00 - 18.00
Thursday, September 21	10.30 - 16.00
Friday, September 22	10.30 - 16.00

SOCIAL PROGRAMME

WEDNESDAY, SEPTEMBER 20		
19.30 - 20.30	**Welcome drinks reception** sponsored by *McAfee Associates*	Plaza Ballroom

THURSDAY, SEPTEMBER 21		
19.30 - 20.00	**Drinks reception**	Plaza Ballroom
20.00 - 01.00	**Gala Dinner** (black tie) Featuring magical entertainment and a Vegas-style casino	Imperial Ballroom

PARTNERS' PROGRAMME

THURSDAY, SEPTEMBER 21		
09.30 - 17.30	**Tour of Boston** (including visits to the *John Hancock Tower*, *Trinity Church*, the *Old South Meeting House*, *Faneuil Hall*, *Quincy Market* and the *Isabella Stuart Gardner Museum*).	Meet in hotel lobby @ 09.15

Partners accompanying delegates to the *VB Conference* are welcome to attend any of the conference dinners and lunches. Tickets may be purchased from the Conference Desk.

Virus Bulletin Conference 1995

Assessment Form

Please help us to plan future conferences by completing the following assessment form. Although it would help us to know your name and company, if you wish to remain anonymous, please do so.

Your form may be handed in to the organisers at the Conference Desk or sent to:

Virus Bulletin Conference, 21 The Quadrant, Abingdon, OX14 3YS, UK

Name ______________________ Company ______________________

1. What was your overall impression of the conference?

❑ Excellent ❑ Good ❑ Average ❑ Poor

Comments ______________________

____

____

2. How did you rate the conference organisation?

❑ Excellent ❑ Good ❑ Average ❑ Poor

Comments ______________________

____

____

3. What was your impression of the Boston Park Plaza and its facilities?

❑ Excellent ❑ Good ❑ Average ❑ Poor

Comments ______________________

____

____

4. Which streams did you attend most often?

❑ Corporate ❑ Technical 1 ❑ Technical 2 ❑ Mixture

000006

5. Of the sessions you attended, how did you rate the speakers?

WEDNESDAY

Jan Hruska - introductory session

Content	❑ Excellent	❑ Good	❑ Average	❑ Poor
Presentation	❑ Excellent	❑ Good	❑ Average	❑ Poor

THURSDAY

Harold Highland

Content	❑ Excellent	❑ Good	❑ Average	❑ Poor
Presentation	❑ Excellent	❑ Good	❑ Average	❑ Poor

Corporate stream

Sarah Gordon

Content	❑ Excellent	❑ Good	❑ Average	❑ Poor
Presentation	❑ Excellent	❑ Good	❑ Average	❑ Poor

Paul Ducklin

Content	❑ Excellent	❑ Good	❑ Average	❑ Poor
Presentation	❑ Excellent	❑ Good	❑ Average	❑ Poor

Jean Hitchings

Content	❑ Excellent	❑ Good	❑ Average	❑ Poor
Presentation	❑ Excellent	❑ Good	❑ Average	❑ Poor

Mike Lambert

Content	❑ Excellent	❑ Good	❑ Average	❑ Poor
Presentation	❑ Excellent	❑ Good	❑ Average	❑ Poor

Wes Ames

Content	❑ Excellent	❑ Good	❑ Average	❑ Poor
Presentation	❑ Excellent	❑ Good	❑ Average	❑ Poor

Technical stream 1

Jonathan Lettvin

Content	❑ Excellent	❑ Good	❑ Average	❑ Poor
Presentation	❑ Excellent	❑ Good	❑ Average	❑ Poor

Neville Bulsara

Content	❑ Excellent	❑ Good	❑ Average	❑ Poor
Presentation	❑ Excellent	❑ Good	❑ Average	❑ Poor

Dmitry Mostovoy

Content	❑ Excellent	❑ Good	❑ Average	❑ Poor
Presentation	❑ Excellent	❑ Good	❑ Average	❑ Poor

Shane Coursen

Content	❑ Excellent	❑ Good	❑ Average	❑ Poor
Presentation	❑ Excellent	❑ Good	❑ Average	❑ Poor

Peter Radatti

Content	❑ Excellent	❑ Good	❑ Average	❑ Poor
Presentation	❑ Excellent	❑ Good	❑ Average	❑ Poor

Scott Gordon

Content	❑ Excellent	❑ Good	❑ Average	❑ Poor
Presentation	❑ Excellent	❑ Good	❑ Average	❑ Poor

Technical stream 2

Frans Veldman					
	Content	❑ Excellent	❑ Good	❑ Average	❑ Poor
	Presentation	❑ Excellent	❑ Good	❑ Average	❑ Poor
Morton Swimmer					
	Content	❑ Excellent	❑ Good	❑ Average	❑ Poor
	Presentation	❑ Excellent	❑ Good	❑ Average	❑ Poor
Jakub Kaminski					
	Content	❑ Excellent	❑ Good	❑ Average	❑ Poor
	Presentation	❑ Excellent	❑ Good	❑ Average	❑ Poor
Ferenc Leitold					
	Content	❑ Excellent	❑ Good	❑ Average	❑ Poor
	Presentation	❑ Excellent	❑ Good	❑ Average	❑ Poor
Igor Muttik					
	Content	❑ Excellent	❑ Good	❑ Average	❑ Poor
	Presentation	❑ Excellent	❑ Good	❑ Average	❑ Poor
David Aubrey-Jones					
	Content	❑ Excellent	❑ Good	❑ Average	❑ Poor
	Presentation	❑ Excellent	❑ Good	❑ Average	❑ Poor

FRIDAY

Corporate stream

Paul Robinson					
	Content	❑ Excellent	❑ Good	❑ Average	❑ Poor
	Presentation	❑ Excellent	❑ Good	❑ Average	❑ Poor
Chris Baxter					
	Content	❑ Excellent	❑ Good	❑ Average	❑ Poor
	Presentation	❑ Excellent	❑ Good	❑ Average	❑ Poor
Judy Edwards					
	Content	❑ Excellent	❑ Good	❑ Average	❑ Poor
	Presentation	❑ Excellent	❑ Good	❑ Average	❑ Poor
Alan Dyer					
	Content	❑ Excellent	❑ Good	❑ Average	❑ Poor
	Presentation	❑ Excellent	❑ Good	❑ Average	❑ Poor
Lucijan Caric					
	Content	❑ Excellent	❑ Good	❑ Average	❑ Poor
	Presentation	❑ Excellent	❑ Good	❑ Average	❑ Poor
Steve White					
	Content	❑ Excellent	❑ Good	❑ Average	❑ Poor
	Presentation	❑ Excellent	❑ Good	❑ Average	❑ Poor

Technical stream 1

Robin Kinney					
	Content	❑ Excellent	❑ Good	❑ Average	❑ Poor
	Presentation	❑ Excellent	❑ Good	❑ Average	❑ Poor

Igor Grebert					
	Content	❑ Excellent	❑ Good	❑ Average	❑ Poor
	Presentation	❑ Excellent	❑ Good	❑ Average	❑ Poor
Pavel Lamacka					
	Content	❑ Excellent	❑ Good	❑ Average	❑ Poor
	Presentation	❑ Excellent	❑ Good	❑ Average	❑ Poor
John Morar					
	Content	❑ Excellent	❑ Good	❑ Average	❑ Poor
	Presentation	❑ Excellent	❑ Good	❑ Average	❑ Poor
Richard Ford					
	Content	❑ Excellent	❑ Good	❑ Average	❑ Poor
	Presentation	❑ Excellent	❑ Good	❑ Average	❑ Poor
Jim Bates					
	Content	❑ Excellent	❑ Good	❑ Average	❑ Poor
	Presentation	❑ Excellent	❑ Good	❑ Average	❑ Poor
Technical stream 2					
Righard Zwienenberg					
	Content	❑ Excellent	❑ Good	❑ Average	❑ Poor
	Presentation	❑ Excellent	❑ Good	❑ Average	❑ Poor
Glenn Coates					
	Content	❑ Excellent	❑ Good	❑ Average	❑ Poor
	Presentation	❑ Excellent	❑ Good	❑ Average	❑ Poor
Roger Riordan					
	Content	❑ Excellent	❑ Good	❑ Average	❑ Poor
	Presentation	❑ Excellent	❑ Good	❑ Average	❑ Poor
Fridrik Skulason					
	Content	❑ Excellent	❑ Good	❑ Average	❑ Poor
	Presentation	❑ Excellent	❑ Good	❑ Average	❑ Poor
Dmitry Gryaznov					
	Content	❑ Excellent	❑ Good	❑ Average	❑ Poor
	Presentation	❑ Excellent	❑ Good	❑ Average	❑ Poor
David Stang					
	Content	❑ Excellent	❑ Good	❑ Average	❑ Poor
	Presentation	❑ Excellent	❑ Good	❑ Average	❑ Poor
Speakers' panel session					
	Content	❑ Excellent	❑ Good	❑ Average	❑ Poor
	Presentation	❑ Excellent	❑ Good	❑ Average	❑ Poor

Comments____

____

____

____

____

____

6. Did you visit the conference exhibition? Yes / No

If 'Yes', how would you rate it as an extension of the conference?

❑ Excellent ❑ Good ❑ Average ❑ Poor

Comments ______________________________

____

____

____

7. What was your impression of the Gala Dinner and entertainment?

❑ Excellent ❑ Good ❑ Average ❑ Poor

Comments______________________________

____

8. How did you hear about the Virus Bulletin Conference? ______________________________

____

9. Do you subscribe to Virus Bulletin? Yes / No

10. What are your impressions of the magazine?

❑ Excellent ❑ Good ❑ Average ❑ Poor

Comments ______________________________

____

____

11. Do you plan to attend the 1996 Virus Bulletin Conference? Yes / No

12. Do you have any suggestions for the location and content of next year's conference?

____

____

____

Thank you for answering this questionnaire

FOREWORD

Proceedings of

The Fifth International Virus Bulletin Conference

000013

0000014

CONTENTS

DAY 2

THE SPEAKERS

WES AMES

Wes Ames is a Senior Principal Scientist in Personal Computer Hardware and Operating Systems for the *Boeing Company* in Seattle, Washington. He is responsible for specific hardware and O/S standards and support in the *Boeing Company*. Over the past six years, Ames has managed the anti-virus activities for *Boeing*, which has responsibility for over seventy thousand personal computers worldwide. These activities range from policy determination to technical support training.

Ames has created the *Boeing* corporate policies necessary for reducing the risk from computer viruses, and is responsible for their implementation and update. He teaches anti-virus classes to technical support analysts, and consults with corporate customers on virus methodologies. He has lectured and led anti-virus discussion groups at the Law Enforcement Conference for Computer Security, and the Society for Information Management.

DAVID AUBREY-JONES

David Aubrey-Jones has been a prolific figure in the computer industry since 1980 and is an authority on viruses and anti-virus warfare. Aubrey-Jones has a PhD from *Leeds University*. In 1988, he started his own company, *Speedlock*, which specialised in copy protection, and soon became a market leader. It was through copy protection that Aubrey-Jones became involved with *Reflex Magnetics*, finally joining as Technical Director in 1991.

Aubrey-Jones is the author of disknet™· *Reflex's* multi-layered computer security solution, which currently protects over 750,000 PCs in multi-nationals, government institutions and other blue-chip organisations worldwide.

JIM BATES

Jim Bates has been involved in electronics all his working life. After service as an Air Radar Engineer in the *Royal Air Force*, he worked as an Electronic Service Engineer on early computers and tabulators. When computer viruses appeared, he was the first in the UK to disassemble them. In 1989, he broke the code encryption and analysed the infamous AIDS Information Disk. This marked the start of his connection with the *Computer Crime Unit* at *New Scotland Yard*: he is now regularly consulted by them and other national and international law enforcement agencies. He runs his own company, *Computer Forensics Ltd*, and he is the designer of the DIBS copying system.

As well as being a respected member of *Virus Bulletin's* advisory board, Bates also belongs to the Computer Security Specialist Group of the *British Computer Society*. He holds a degree in Electronic Engineering, and was elected a Fellow of the *Institute of Analysts and Programmers* in 1987. He was appointed President of the ruling council of the *IAP* in 1993, and is an active member of the *Forensic Science Society*.

000017

NEVILLE BULSARA

Neville Bulsara first began programming in assembler at the age of 18. In 1988, at the age of 21, he pioneered the anti-virus movement in India by being the first in the country to take apart the Brain virus and write an antidote for the same. Since then, he has been at the forefront of the battle against viruses in India.

Since 1989, Bulsara has served as a consultant on the field to the Government of India, some of the largest corporations in the country and various defence establishments. He considers the lack of user-awareness as the greatest factor contributing to the virus menace, and so is to be regularly found lecturing on the subject at user-group meetings and computer shows.

LUCIJAN CARIC

Lucijan Caric gained his LL.B. at the *University of Zagreb*. When he started to work in the area of computer security and anti-virus measures in 1991, he already had extensive experience in the computing. He joined *United Nations Peace Forces (UNPF)* in the former Yugoslavia in 1993 and is currently the Special Projects Coordinator in the Information Technology Services Section. Caric is responsible for computer security projects and the development of major projects conducted by the section.

In an effort to improve standards of computer security and anti-virus protection in Croatia, Caric is also acting as a contributing editor to a leading Croatian computer magazine, *Bug*, and has taken part in a series of broadcasts about computer viruses on the metropolitan TV station.

GLENN COATES

Glenn Coates is 23 years old and has recently completed a BSc (Hons) degree in Computing Science at *Staffordshire University*. For his final year project, he developed a prototype Virus Description Language (VDL) upon which his paper at *VB '94* was based.

He is now working at *Security Information Systems Ltd* (*SISL*) as a trainee security evaluator. His other computing interests include operating systems design, compiler writing, neural networks and human computer interaction. His hobbies include fitness training, parachuting and socialising.

PAUL DUCKLIN

Paul Ducklin's involvement in the anti-virus field started in 1989 in South Africa, at the time that computer viruses first began to appear there. He spent five years as the head of the Computer Virus Lab at the South African *Council for Scientific and Industrial Research* in Pretoria, before moving to England earlier this year to join the anti-virus team at *Sophos Plc*, the producers of SWEEP. Though a recent arrival in the EC, he has attempted to make his mark as a true European by eating British cheese, driving a French car, and riding an Italian motorcycle.

ALLAN DYER

Allan Dyer first studied biological viruses, graduating in Microbiology from *University College*, London, in 1984. He switched fields, gaining a Master's in Control Engineering from *Sheffield University* in 1987 and combined his skills programming in a haematology research laboratory. He first met computer viruses in 1988 while a Systems Programmer at the *London School of Hygiene and Tropical Medicine*, and joined the team controlling them in a number of London colleges. He moved to Hong Kong in 1993, and now manages F-PROT Professional for *Yui Kee Co. Ltd*.

JUDY EDWARDS

Judy Edwards is a Microcomputer Software Specialist at *Illinois State University*, where she is a member of the WWW Team and an ftp site administrator. She provides Internet training and help desk support to faculty, staff and instructional computer labs, and also does independent Internet consulting.

Edwards holds a Master's degree in Instructional Systems Technology from *Indiana University's* College of Education, and a Personal Computer Coordinators certificate from the *University of Southern Maine*.

RICHARD FORD

Richard Ford obtained a BA in Physics from Queen's College, Oxford, in 1989, and went on to study for a D. Phil. in Semiconductor Physics. His interest in computer viruses began during the course of his research, when the computer he was using became infected with the Spanish Telecom virus. The virus triggered, nearly destroying six months' worth of results, but rather than turning to anti-virus software for the answer, Ford analysed the virus himself.

In the following year, he wrote various articles for *Virus Bulletin* and became editor in January 1993. He has since lectured and talked world-wide on the problems posed by malicious software. In April of this year he joined the *National Computer Security Association (NCSA)* in the US as Director of Research.

SARAH GORDON

Sarah Gordon, Security Analyst for *Command Software Systems, Inc*, has been an invited speaker at such diverse conferences as those sponsored by *The American Association for the Advancement of Science, EICAR, DEFCON*, and *Virus Bulletin*. A frequent contributor to security industry technical publications, she is also the winner of the Sec 94 IFIP TC11 award for her research on social and ethical implications of technology: 'Technologically Enabled Crime: Shifting Paradigms for the Year 2000'.

She has recently completed research projects at *Indiana University* in Unix Security and in Computer Ethics. Current projects include 'Development of Information Security Education in Developing Countries', and 'Anti-Virus Product Certification Methodologies'. Sarah can be heard at the upcoming *National Computer Security Conference* in Baltimore, in October and also at *Compsec* in London, in November.

SCOTT GORDON

Scott Gordon is the Product Manager for *McAfee Associates*' security solutions. Acting as the focal point relating to product development, positioning, delivery and support, he is a company spokesperson and among resident experts on issues relating to computer viruses, enterprise date quality assurance, and security. Gordon's background in the computer industry spans a broad range of experience; from retail, channel and corporate sales to consulting and product marketing / development manager for network security and management.

Prior to joining *McAfee*, Gordon was the product development manager for network security and management products at *Cheyenne Software*. Before this, he was responsible for product marketing and technical sales functions with *Computer Associates International* and, previously, was a product manager with a network technology seminar company and an independent systems consultant. He has an MBA from the *University of Phoenix* and a BBS from *Hofstra University*.

IGOR GREBERT

Igor Grebert studied in Paris and graduated in 1989 from *Ecole Centrale de Paris*, with a major in bio-technology. He worked as a post-doctorate researcher at Stanford designing neural networks applied to target tracking, image analysis and automatic pilots. In 1990, he started to use his pattern matching expertise to help detect the growing number of computer viruses. In 1993, he headed the redesign of *McAfee's* VirusScan product line as manager of the research and development team. Today he applies his skills to designing enhanced anti-virus systems, integrating his years of experience in the field.

DMITRY GRYAZNOV

Dmitry Gryaznov was born in 1961 in Frunze, Kirghyzskaya SSR, USSR. He was educated at the *Moscow Skills Improvement Institute* and the *Moscow Physics and Technology Institute (MPhTI)*, where he gained a Unix Operating System programmer and administrator certificate and an MSc in Computer Science in 1984.

Since graduating, Gryaznov has held various positions in the *Program Systems Institute* at the *Russian Academy of Sciences*. Earlier this year he moved to the UK, and now works as a senior virus research analyst with *S&S International Plc.*

HAROLD HIGHLAND

Dr. Harold Joseph Highland, FICS, FACM, is the only Fellow of both the *Irish Computer Society* and the *US Association for Computer Machinery*. The career of this 'elder statesman' of computing spans over 57 years with experience in the military, industry and academia.

His professinal life started when he was designated as Honor Graduate of his military class and commissioned on his college graduation in 1938. He served as Provost Marshall and was seconded to cryptographic analysis and later to intelligence analysis. In addition to working for *The New York Times* and other newspapers, Highland was a research statistician, an economist, a management consultant, a methods engineer, a magazine editor and publisher, a television producer and even MCed one of his programs. He also owned an advertising/public relations organization, was a dean of a graduate school, associate dean of a liberal arts college, director of various computer centers, a consultant, and a classroom teacher. Likewise, he has worked with various government agencies and even today serves as computer security consultant to the Beijing government.

Prior to his retirement in 1981, Highland planned a new international journal, *Computers & Security*, the first issue of which appeared in 1982: he was Editor-in-Chief. In 1984, his publication became the official journal of *International Federation for Information Processing's Technical Committee 11* on information security [IFIP/TC11]. Dr. Highland is a prolific author, who has written 27 books in the past 35 years. He has also published and/or presented over 200 technical papers in various areas of computing at regional, national and international conferences, as well as in professional journals.

JEAN HITCHINGS

Jean Hitchings obtained a BSc in Computer Science from the *University of Westminster* in 1982 and went on to study for an MSc in the same subject at the *University of London*. Since January 1992 Jean has been a lecturer in Information Technology at the *University of Nottingham*. She has recently gained a PhD in Computer Science from the *University of East Anglia*.

JAN HRUSKA

Jan Hruska is the Technical Director of *Sophos Plc* in Oxford. A graduate of Downing College, Cambridge, he gained his doctorate at Magdalen College, Oxford. In April 1980, he formed *Sophos* with Dr. Peter Lammer as a computer design partnership: the company was incorporated in 1987 and specialises in data security. He is a co-author (with Dr Keith Jackson) of 'The PC Security Guide', published by *Elsevier*, and 'Computer Security Solutions', published by *Blackwells*. He is the author of 'Computer Viruses and Anti-Virus Warfare', published by *Ellis-Horwood*. Hruska regularly speaks at computer security conferences and consults on a number of security aspects, including virus outbreaks. His extra-curricular interests include flying, skiing, scuba-diving and piano-playing and he is an ex-member of *Mensa*.

JAKUB KAMINSKI

Jakub Kaminski graduated and received an MSc in Electronics from *Warsaw Technical University* in 1986. He went on to work for the *Institute of Fundamental Technological Research*, at the *Polish Academy of Sciences*, spending most of his time doing system programming and working in different assemblers.

In 1992, Kaminski moved to Australia and started working for *CYBEC*. He disassembles new viruses and incorporates them into *CYBEC*'s VET. In June 1995, he joined the *Virus Bulletin* team as Technical Editor.

ROBIN KINNEY

Robin Kinney has dedicated nearly his entire career to devices used for treating cancer. The last nine years he has spent in software management of *Varian Oncology Systems*, where he has managed both departments and projects.

Kinney is an advocate of software process improvement. He led the effort within *Varian Oncology Systems* for ISO-9001 certification, and has spent more than a year as chair of the Software Process Improvement Committee within that organization.

PAVEL LAMACKA

Pavel Lamacka graduated in 1971 and, in 1983, gained a degree in computer science from the *Slovak Technical University* in Bratislava. He has worked at several research institutions, mainly in software engineering, taking part in work on software - including a real-time operating system for the first Slovak control computer. He was a member of the team which designed and implemented the BPS programming system based on a MODULA-2-like programming language (this system was used for years on *IBM* mainframes and *DEC* minicomputers) and has also worked on a perspective block-building programming system.

In Spring 1988, he encountered and disassembled his first virus and gave his first lecture on viruses and other computer infiltration means. Since then, he has been active in computer security, initially as a consultant, later as an author of computer security products. Lamacka is currently the Head of the *Computer Accidents Research Center*, which he formed in 1992 and which is based in *HTC*, a large private computer company.

MIKE LAMBERT

Mike Lambert is Electronic Security Manager at *Frontier Corporation*. He has been involved with computer viruses since 1988; doing analyses, testing products and assisting in cases of virus infections. He has written other papers on Disaster Recovery Disks for the PC and fatal DOS vulnerability.

FERENC LEITOLD

Ferenc Leitold graduated from the Department of Informatics at *Budapest Technical University*. Having completed a three year post-graduate course at the university, he is now in the process of doing a PhD on the mathematical modelling of operating systems and computer viruses.

Leitold joined the fight against computer viruses in 1988, with the appearance of the first virus in Hungary (Cascade). He is a founding member of the *Hungarian VirusBuster Team*, operating under the aegis of *Hunix Ltd.* Their anti-virus product, VirusBuster for DOS and NetWare, is sold throughout Hungary.

JONATHAN LETTVIN

Jonathan Lettvin has been *Lotus*' anti-virus principal investigator for six years. He developed all of the company's anti-virus policies and procedures. He also created the *Lotus* 'Release Engineering Anti-virus Laboratory' (REAL): this is responsible for examining all *Lotus* products for viruses before shipping. REAL has an unblemished record of preventing viruses being shipped in *Lotus* products.

Lettvin has been programming for many years, and views his anti-virus work at *Lotus* as strongly influenced by his training at *MIT*, medical school and *Bell Labs*, as well as some beneficial professional partnerships.

OverByte was incorporated to develop and market products based on Lettvin's experience fighting viruses in the *Lotus* corporate environment. *Lotus* has been generous in granting all commercial rights for DisQuick/ViRemove to *OverByte* and is its first and best customer.

DMITRY MOSTOVOY

Dmitry Mostovoy was born in 1962, in Moscow. He graduated from the *Moscow Aviation Institute*, specializing in Space Science, and then worked at the *Keldysh Institute of Applied Mathematics* at the *Russian Academy of Sciences*, on the dynamics of the re-entry of space vehicles. Mostovoy participated in the Russian orbiter 'Buran' project.

Mostovoy obtained a PhD degree in theoretical mechanics and, in 1989, became interested in the computer virus problem. Since 1991, he has been a leading anti-virus designer at *DialogueScience Inc*, and the author of one of the renowned Russian anti-virus utilities, ADinf, a data integrity checker. Mostovoy is also an active yachtsman.

IGOR MUTTIK

Muttik Igor was born in Moscow in 1962. He graduated from the Physics Department of *Moscow State University* in 1985, where he subsequently worked on low temperature physics and used computers in physics experiments. In 1989, he received a PhD in physics and mathematics from *Moscow University*. He then worked on the use of computers in education and experiments, and published more than 50 scientific articles in various Russian and international magazines. In 1988, he became interested in computer viruses, although this anti-virus activity was just a hobby.

A programmer and a researcher, Muttik has developed an interest in the fundamental investigation of viruses. He is especially engaged in complex polymorphic, armored and multi-partite viruses. In 1994 he joined *CARO*. In August 1995 he was appointed Virus Research Analyst at the Virus Laboratory of *S&S International Plc*, in Aylesbury, UK.

PETER RADATTI

Pete Radatti is the founder and President of *CyberSoft, Inc*, manufacturers of VFind, the antivirus software product which executes on Unix systems. VFind simultaneously scans for Unix, MS-DOS, Macintosh and Amiga destructive software while providing cryptographic integrity to filesystems.

ROGER RIORDAN

Roger Riordan graduated in Electrical Engineering from *Melbourne University* in 1954. After two years with *English Electric* in the UK, and some years with *CSIRO*, he set up *CYBEC Electronics* in 1973. At *CYBEC*, he designed a wide range of scientific and industrial equipment. He joined *Chisholm Institute of Technology* as a lecturer in Electronics in 1983, and became involved with computer viruses in 1989, when the PC labs were paralysed by an outbreak of the Stoned virus. He wrote the first version of VET to counter it, and gave it to the students as shareware.

Riordan has attended a number of international conferences, and published several papers on his work related to virus research. He is a member of *CARO*, the international anti-virus research organisation.

PAUL ROBINSON

Paul Robinson, Editor of *SECURE Computing* magazine, has a long track record writing about security issues and related solutions. Prior to assuming the editorship, he wrote for many of the top UK computer and business magazines. *SECURE Computing* is an international security journal with one of the largest circulations of a publication in its field.

FRIDRIK SKULASON

Fridrik Skulason received a BSc from the *University of Iceland*. In 1987, he started his own software company in Reykjavik, specialising in programs tailored for Icelandic needs. Skulason became involved in computer viruses in early 1989, when they first appeared in Iceland. He is the author of F-Prot anti-virus software and is a former Technical Editor of *Virus Bulletin*.

DAVID STANG

David Stang has been involved in computer security for several years, and is currently the Chief Technical Officer for *Norman Data Defense Systems, Inc.* He was the founder of the *National Computer Security Association* (*NCSA*) and also founder and chairman of the *International Computer Security Association* (*ICSA*), the umbrella organization for the *NCSA*. He is the author of several books on computer security including *Norman*'s 'Computer Virus Handbook', and co-author (with Syliva Moon) of 'Network Security Secrets'. Stang edited *Virus News and Reviews* (*VNR*), a journal which was published monthly throughout 1992. He is also a member of the editorial board and columnist for *InfoSecurity News*, and has contributed over 160 articles to the computer trade press.

Stang holds a PhD from *Syracuse University*, an MS from the *University of Toronto* and a BS from *Cornell University*.

MORTON SWIMMER

Morton Swimmer was born in New York City, USA. After moving to Germany, he studied first in England and then at the *University of Hamburg*, Germany. He is currently close to completing his Master's degree in Computer Science (Informatik).

Swimmer has been a member of the Virus Test Center at the *University of Hamburg* since its inception in 1988. He has also managed *S&S International (Deutschland) GmbH* as well as working in the Virus Lab at *S&S International Plc*, UK. His research interests are in computer and network security, in particular computer viruses and worms.

IAN WHALLEY

Ian Whalley has been Editor of *Virus Bulletin* since April 1995; before that he worked at *Sophos Plc* developing an anti-virus solution for Windows NT. He is a graduate of *Manchester University* (1994), where he studied Physics and Computer Science, and it was here that he first became interested in the field of computer security. He maintains a keen interest in viruses on the new generation of PC operating systems, not least Windows NT.

STEVE WHITE

Steve White received a PhD from *UCSD* in theoretical physics in 1982, and since then has been at the *IBM Thomas J. Watson Research Center*. He has had articles published on a variety of subjects, including condensed matter physics; optimization by simulated annealing; software protection; computer security and computer viruses. White holds several patents in security-related fields. He organized and now manages the High Integrity Computing Laboratory at *IBM Research*, where he is responsible for the research and development of *IBM* anti-virus products. His research interests include the long-term implications of computer viruses and other self-replicating programs in distributed systems.

RIGHARD ZWIENENBERG

Righard Zwienenberg is the Research & Development Manager of *Computer Security Engineers Ltd*. He started dealing with computer viruses in 1988 after encountering the first virus problem on a system at the *Technical University of Delft*. His interest thus kindled, Zwienenberg has studied virus behaviour and presented solutions and detection schemes ever since - initially as an independent consultant and later, in 1991, with *CSE*. His interests have now broadened to include general security issues, such as network protection and internet firewalls.

THE DELEGATES

(as of 29/08/95)

A

Emmanuel Areola	EBA Communications	UK

B

Sqn Ldr Mark Baker	RAF High Wycombe	UK
Philip Bancroft	Digital Equipment Corporation	USA
Pavel Baudis	Alwil Software	Czech Republic
Ken Bauman	Computer Security Consultants Inc	USA
Richard Beard	State Street Bank	USA
Juergen Benz	Deutsche Telekom AG	Germany
Joseph Bernfeld	Merrill Lynch	USA
Daphne Bertrand	United Parcel Service	USA
Derril Bibby	Texaco Group Inc	USA
Robin Bijland	ESaSS GmbH	The Netherlands
Pat Bitten	S&S International	UK
Jim Blackwell	US Department of Agriculture	USA
Peter Bohm	NoVIR Data	Germany
Vesselin Bontchev	Frisk Software International	Iceland
Kevin Bosworth	British Telecom	UK
Adrienne Botti	Department of the Navy	USA
Donald Boyd	New York Times	USA
Carl Bretteville	Norman Data Defense Systems	Norway
James Brown	Fidelity Investments	USA
Charles Brown	Keiretsu Institute	USA
Torri Buchwald	Pratt & Whitney	USA
John Butler	The Automobile Association	UK

C

Bob Cartwright	Chevron	USA
Banda Casella	Royal Bank of Canada	Canada
Alex Chen	Cheyenne Software Inc	USA
Sing Bin Chew	TAS Inc	USA
Graham Cluley	S&S International	UK
Gary Cornelius	Command Software Systems	USA

D

Martin Damen	Ministry of Defence	UK
Helen Dawe	Sophos Plc	UK
Chris Debracy	Command Software Systems	USA
Paul Docherty	Portcullis Computer Security	UK
Moti Dover	Eliashim Microcomputers Inc	USA
Dyan Dyer	Command Software Systems	USA

F

Wesley Fagan	US Army	USA
Tom Farrell	Alternative Computer Technology	USA
Cheryl Flerk	Detroit Edison	USA
Thomas Le Fleur	S&S International	UK
David Flury	BT Payphones	UK
Dan Fox	Defense Logistics Agency	USA
Susan Franco	American Airlines	USA

G

Blase Gaude	Sandia National Laboratories	USA
Mr Donny Gilor	Iris Software	Israel
Ray Glath	RG Software Systems	USA
Eint Goedhart	Ministry of Defense	The Netherlands
Albert Gorter	NATO / NAPMA	The Netherlands
James Gosler	Sandia National Laboratories	USA
Paul Graham	Bureau of Reclamation	USA
Jeremy Gumbley	Command Software Systems	USA
Pege Gustafsson	Telia AB	Sweden

H

Ronald Halbgewachs	Sandia National Laboratories	USA
Neil Halliday	Sophos Plc	UK
Cynthia Hanlon	Fidelity Investments	USA
Philip Harris	Fidelity Investments	USA
Mike Hills	Ministry of Defence (Army)	UK
Robert Hinten	Environmental Protection Agency	USA
Richard Ho	IBM Corporation	USA
Steve Holstein	Virus Bulletin	USA
Tony Hopkins	Kingston University	UK
Zoltan Hornak	Technical University of Budapest	Hungary
Frank Horwitz	Reflex Inc	USA
Tore Hoyem	Norske Shell A/S	Norway
Mikko Hypponen	Data Fellows Ltd	Finland

J

Craig Jackson	Datawatch Corporation	USA
Portia Jackson	Department of Veterans Affairs	USA
Richard Jacobs	Sophos Plc	UK
Ken Jaworski	Detroit Edison	USA
Joy Johnson	Intel	USA
Martin Jones	Computacenter	UK

K

Natalya Kasperskaya	KAMI	Russia
Norkio Kato	Jade Corporation Ltd	Japan
Tapio Keihanen	MikroPC Magazine	Finland
Greg Kendig	AMP Incorporated	USA
Jeffrey Kephart	IBM	USA
Michal Kovacic	Alwil Software	Czech Republic
Eduard Kucera	Alwil Software	Czech Republic

L

Lawrence LaBella	Merrill Lynch	USA
Paul Lawrence	S&S International	UK
Orlton Lawrence	Toronto Dominion Bank	Canada
Dave Leigh	Staffordshire University	UK

Darren Leonard	Alternative Computer Solutions	USA
Sharon Lettvin	OverByte Corporation	USA
Myron Lewis	Norman Data Defense Systems Inc.	USA
Vince Lombardi	Fleet & Industrial Supply Center	USA
Tina Lombardi	McAfee Associates	USA
Greg Lutz	OverByte Corporation	USA
Merrill Lynch	Chevron	USA
Sherry Lynch	Detroit Edison	USA

M

Jafar Muhammad Mamun	El-Mamun Ent.	Ghana
Henry Matos	The Segal Co.	USA
Jack McAulay	University of Eninburgh	UK
Tom McCllough	IBM Corporation	USA
Svein Meland	Allianse Informasjonssystemer	Norway
Alison Millar	Standard Life Assurance Co	UK
Mahesh Moorthy	IBM Corporation	USA
Seiji Murakami	Jade Corporation Ltd	Japan
Akihiko Muranaka	Jade Corporation Ltd	Japan

N

Sean Nabeau	ESaSS GmbH	The Netherlands
Carey Nachenberg	Symantec	USA
Kurt Natvig	Norman Data Defense Systems	Norway
Senthil Nathan	IBM	USA
Kevin Norris	Allied Irish Bank Group	Ireland

O

Martin Odvarko	Alwil Software	Czech Republic
Jorgen Olsen	DOU, Odense University	Denmark
Ernst Oud	Crypsys	The Netherlands
Ian Overton	GPT Ltd	UK

P

Therese Padilla	Command Software Systems	USA
Charles Parker	IBM	USA
Keith A Peer	Central Command Inc.	USA

Chen Yi-Pen	Trend Micro Devices Inc	Taiwan
Manfred Philipowsky	Deutsche Telekom AG	Germany
Donald Phipps	The Clorox Company	USA
Andre Pitkowski	Compusul	Brazil
Kirstin Police	Alternative Computer Solutions	USA
Alastair Port	KPMG	UK
Edward Pring	IBM	USA
Judy Pruitt	Alternative Computer Technology	USA
Richard Ramza	Deere & Company	USA

R

Francisco Ramos	Microasist	Mexico
Carole Reid	Defense Commissary Agency	USA
Charles Renert	Symantec	USA
Sally Riordan	Cybec Pty Ltd	Australia
Eduardo Rios		Mexico
Frank Roache	BBC World Service	UK
Jake Roddy	Defense Contracts Audit Agency	USA
Rhonda Rosenbaum	IBM	USA
Marvin Ruppert	National Futures Association	USA

S

Alla Segal	IBM	USA
Marek Sell	Apexim Co S.A	Poland
James Shaeffer	Reflex Inc	USA
Lee Jieh-Sheng	Trend Micro Devices Inc	Taiwan
Risto Siilasmaa	Data Fellows Ltd	Finland
Lester Simons	Lloyd's Register	UK
George Sneddon	Sophos Plc	UK
Alan Solomon	S&S International	UK
Susan Solomon	S&S International	UK
David Stanley	Royal Air Force	UK
Philip Statham	CESG/GCHQ	UK
Ken Stieers	Ontrack Computer Systems	USA
Howard Stone	BBC World Service	UK
Tom Stormer	Alternative Computer Solutions	USA

Robert Stroud	Cybec Pty Ltd	Australia
Mary Sullivan	Alternative Computer Technology	USA

T

Gabriel Takami	S&S International	Mexico
Allen Taylor	Customs Service	USA
Tony Tedridge	Minsitry of Defence (Army)	UK
Howard Thaw	ESaSS GmbH	The Netherlands
Peter Theobald	Quantum System Software	India
Andrew Tilling	Royal Military Police Info Systems	UK
Shelagh Todd	NationsBank Services Inc	USA
Ruben Tovar	S&S International	Mexico
Howard Townsend	Virginia Housing Development Auth.	USA
Alan Tremblay	Statistics Canada	Canada

V

Gert van de Nadort	Crypsys	The Netherlands
Chan Vo	The New York Times Company	USA

W

Reed Ward	Ontrack Computer Systems	USA
Simon Webber	Defense Research Agency	UK
Joe Wells	IBM	USA
Ian West	Royal Air Force	UK
Simon John Williams	Bass Brewers Ltd	UK
Denis Woods	Renaissance Contingency Services	Ireland
Simon Woolley	Sophos Plc	UK

EXHIBITORS

Alwil Software

Command Software Systems

Computer Security Consultants Inc

Cybec Pty Ltd

Eliashim Microcomputers Inc

ESaSS GmbH

IBM Corporation

McAfee Associates

Norman Data Defense Systems

Ontrack Computer Systems

Reflex Inc

RG Software Systems

S&S International

Sophos Plc

Virus Bulletin Ltd

ANTI-VIRUS PRODUCT DEVELOPERS

The following is a list of anti-virus product developers known to Virus Bulletin in August 1995. Its accuracy and currency are not attested and cannot be guaranteed.

Alwil Software, Prubezna 76, CS-100 00 Prague 10, Czech Republic
Tel +42 278 22050, Fax +42 278 22553
Product: *Avast!*

Carmel Software Engineering, PO Box 25055, Hamachshev Ltd Hahistradrut Av 20, Haifa, Israel.
Tel +972 4 416976, Fax +972 4 416979
Product: *TNT Anti-virus, Carmel Anti-virus*

Cheyenne Software Inc., 3 Expressway Plaza, Roslyn Heights, New York 11577, USA.
Tel +1 516 629 4459, Fax +1 516 484 3446
Product: *Inoculan*

Command Software, Suite 500, 1061 E Indian Town Road, Jupiter, FL 33477, USA.
Tel: +1 407 575 3200, Fax +1 407 575 3026
Product: *F-PROT*

Computer Security Engineers Ltd, Postbus 85 502, 2508 CE Den Haag, The Netherlands.
Tel +31 70 362 2269, Fax +31 70 365 2286
Product: *PC Vaccine Professional*

Cybec Pty. Ltd., 133 Alexander Street, Crows Nest, NSW 2065, Australia.
Tel +61 2 9965 7216, Fax +61 3 2438 2335
Product: *VET*

Cybersoft, 1508 Butler Pike, Conshohocken, PA 19428, USA.
Tel: +1 610 825 4748, Fax: +1 610 825 6785
Product: *V-FIND*

Data Fellows Ltd, Paivantaite 8, 02210 Espoo, Finland.
Tel: +358 0 478 444 , Fax: +358 0 478 44599
Product: *F-PROT*

Datawatch Corporation, 234 Ballardvale Street, Wilmington, MA 01887, USA.
Tel: +1 508 988 9700, Fax: +1 508 988 0105

Product: *Virus X Screen Link, Virex for Macintosh*

Diamond Chip Computers CC, 2nd Floor, 4 Susmann Avenue, Blairgowrie, Randburg, Johannesburg, South Africa.
Tel: +27 11 886 3131, Fax: +27 11 86 3331

Product: *ViruGuard*

EliaShim Microcomputers Ltd., PO Box 9195, Haifa 31091, Israel.
Tel: +972 4 516111, Fax: +972 4 528613

Product: *Virusafe*

EMD Enterprises, 6 Cardinal Drive, Glenrock, PA 17327, USA.
Tel: +1 717 235 4261, Fax: +1 717 235 1456

Product: *EMD Armor Plus*

ESaSS BV, Saltshof 10-18, 6604 EA Wijchen, The Netherlands.
Tel: +31 8894 22282, Fax: +31 8894 50899

Product: *Thunderbyte*

Frisk Software International, PO Box 7180, 127 Reykjavik, Iceland.
Tel: +354 1 617273, Fax: +354 1 617274

Product: *F-PROT*

H+BEDV Datentechnik GmbH, Olgastrasse 4, D-88069 Tettnang, Germany.
Tel: +49 7542 93040, Fax: +49 7542 52510

Product: *AV Scan*

Hunix Ltd, Budafoki ut. 57/A, 1111 Budapest, Hungary.
Tel: +36 1 186 7408, Fax: +36 1 186 7408

Product: *Virus Buster*

IBM, TJ Watson Research Centre, PO Box 218, Route 134, Yorktown Heights, NY 10598, USA.
Tel: +1 914 945 3000, Fax: +1 914 945 2141

Product: *IBM AV*

Information Security Services Inc., 1211 Distribution Way, Beltsville, MA 20705, USA.
Tel: +1 301 470 2500, Fax: +1 301 470 2507

Product: *Detect Plus*

Intel Corp., 5200 N E Elam Young Parkway, Hillsborough, OR 97124, USA.
Tel: +1 503 629 7354, Fax: +1 503 629 7580

Product: *LAN Desk Virus Protect*

Iris Software, 6 Hamavo Street, Givataim 53303, Israel.
Tel: +972 3 571 5319, Fax: +972 3 318 731

Product: *Iris AV*

Leprechaun Software Pty. Ltd., 75 Redland Bay Road, Capalaba, Queensland 4157, Australia.
Tel: +61 7 823 1300, Fax: +61 7 823 1233

Product: *Virus Buster*

McAfee Associates, 2710 Walsh Avenue, Suite 200, Santa Clara, CA 95051-0963, USA.
Tel: +1 408 988 3832, Fax: +1 408 970 9727

Product: *Virus-Scan, V-Shield*

Norman Data Defense Systems, Suite 201, 3028 Javier Road, Fairfax, VA 22031, USA.
Tel: +1 703 573 8802, Fax: +1 703 573 3919

Product: *Norman Virus Control*

Panda Systems, 801 Wilson Road, Wilmington, DE 19803, USA.
Tel: +1 302 764 4722, Fax: +1 302 764 6186

Product: *Panda Pro, Bear Lock, Dr Panda Utilities*

Peter Hoffmann Service GmbH, Friedrichsplatz 12, 68165 Mannheim, Germany.
Tel: +49 621 4311 901, Fax: +49 621 444 273

Product: *PC Safe*

Reflex Magnetics Ltd, 31-33 Priory Park Road, Kilburn, London, NW6 7OP, UK.
Tel: +44 71 372 6666, Fax: +44 71 372 2507

Product: *DiskNet*

RG Software Systems, 6900 East Camelback, Suite 630, Scottsdale, AZ 85251, USA.
Tel: +1 602 423 8000, Fax: +1 602 423 8389

Product: *Vi-spy*

Safetynet Inc., 140 Mountain Avenue, Springfield, NJ 07081, USA.
Tel: +1 908 851 0188, Fax: +1 908 276 6575

Product: *VirusNet-Pro*

SA Software, 28 Denbigh Road, London, W13 8NH, UK.
Tel: +44 81 998 2351, Fax: +44 81 998 7507

Product: *PC Immunize II*

S&S International Plc, Alton House, Gatehouse Way, Aylesbury, Bucks HP19 3XU, UK.
Tel: +44 296 318700, Fax: +44 296 318777

Product: *Dr Solomon's Toolkit*

SikkerhedsRadgiverne ApS, Knabrostraede 20, 4th Floor, DK-1210 Copenhagen K, Denmark.
Tel: +45 3332 3537, Fax: +45 3332 3547

Product: *ASP Integrity Toolkit*

Softcraft AG, Niederwiesstrasse 8, CH-5417 Untersiggenthal, Switzerland.
Tel: +41 56 28 11 16, Fax: +41 56 28 11 16

Product: *VIP*

Sophos Plc, 21 The Quadrant, Abingdon Science Park, Abingdon, Oxfordshire OX14 3YS, UK.
Tel: +44 1235 559933, Fax: +44 1235 559935

Product: *Sweep, Vaccine*

Symantec Corporation, 10201 Torre Avenue, Cupertino, CA 95014, USA.
Tel: +1 408 725 2762, Fax: +1 408 253 4992

Product: *Norton Anti-virus, CPAV*

The Davidson Group, 20 Exchange Place, 27th Floor, New York, NY 10005, USA.
Tel: +1 212 480 1050, Fax: +1 212 422 1953

Product: *Vaccine*

Thompson Security Software, PO Box 669306, Marietta, GA 30066, USA.
Tel: 0101 404 971 8900, Fax: +1 404 971 8828

Product: *Doctor*

Trend Micro Devices Inc., 1F #28 Li-Shui Street, Taiwan, Republic of China.
Tel: +886 2 312 0191, Fax: +886 2 341 2137

Product: *PC-Cillin, Stationlock*

Visionsoft, Unit C7, Enterprise Way, Five Lane Ends, Idle, Bradford, West Yorkshire BD10 8EW, UK.
Tel: +44 274 610503, Fax: +44 274 616010

Product: *SmartScan, Immunizer*

D
A
Y

1

000037

THE ANTI-VIRUS STRATEGY SYSTEM

Sarah Gordon

Command Software Systems, Inc, 1061 E. Indiantown Road, Suite 500, Jupiter, FL 33477, USA
sarah@dockmaster.ncsc.mil

ABSTRACT

Anti-virus protection is, or should be, an integral part of any Information Systems operation, be it personal or professional. However, our observation shows that the design of the actual anti-virus system, as well as its implementation and maintenance, can range from haphazard and sketchy to almost totally nonfunctional.

While systems theory in sociological disciplines has come under much attack, it has much to offer in the management of integration of technological applications into daily operations. We will examine the 'anti-virus' strategy (Policy, Procedure, Software [selection, implementation, maintenance]), focusing on areas where the 'system' can fail. We will address this interaction from a business, rather than a personal computing, point of view.

The Anti-Virus Strategy System will examine anti-virus strategies from a Holistic General Systems Theory perspective. By this, we mean that we will concern ourselves with the individual parts of the system, their functionality, and their interaction. We will draw from various IT models specifically designed to provide a holistic, forward-thinking approach to the problem, and show that for our strategy to flourish, we must concern ourselves with the system as a whole, not merely with its individual components.

1 INTRODUCTION

Computer virus. System failure. These words bring to mind a computer system brought to its knees - data corrupted and time wasted. Is this an accurate picture? We hear arguments against investing in virus protection: 'Viruses are mythical. Your chances of getting hit by one are pretty rare.' Others tell us anti-virus software is a necessity: 'Viruses can cost your company a lot of money. Better safe than sorry.' What are we to believe?

Let's assume that you don't have any anti-virus software. If you are 'hit' by a virus, the cost will be proportional to the value of your data and the value of your time. Independent studies [1] have shown that this cost can be quite high, depending on these factors as well as environmental factors such as how many computers you have (Note: If your data is of little or no value, and if your time is worthless, then you can well afford not to have an anti-virus strategy).

We will assume here that your data is worth something to your company, and that your time also has a significant value. In this case, you will want to protect your computer system from viruses. We will concede for the purists among us that not all viruses are intentionally harmful, but stipulate that intentional harm is not requisite for actual harm. For our purposes, allocating disk space and CPU time and/or modification of files without knowledge and consent (implied or otherwise) constitutes damage, as do deliberate or unintentional disruption of work, corruption of data and the lost time mentioned earlier. Basically, we are saying viruses are bad and we want to protect against them (there may be some wonderful new virus out there in development that can help us, but that is beyond the scope of this paper).

Fortunately, we are in luck. The very thing we need already exists: software, which will detect 100 percent of viruses listed by the Wildlist [2] as being known to be in the wild. In tests run against a library matched with the Wildlist, several programs were capable of detecting all such viruses. The necessity of detection of 'lab' viruses is another matter, and will not be covered at this time, although it is addressed in [3].

Since we have such software, we should have no problems. However, there are problems. Something is wrong. Before examining the sources of the problem, a few comments on definitions we will be using are in order.

2 DEFINITIONS

The definitions used here are pretty generic, and are adapted for use in an interdisciplinary approach to the problems addressed. Some among us would argue that the systems movement was born out of science's failures [4], but in this paper, we take the view that General System theory is a child of successful science, and as most children, it sees things through optimistic eyes. We have specifically avoided in-depth discussion of categorical schemes, generalizations, and other commonly used 'tools' of General Systems thought, and have focused instead on the simplest of the simple. The ideas in this paper are drawn heavily from very basic works in systems theory. They are not new ideas, but it is our hope that their application to the management of security and computer viruses will help us identify some of the problems we may be overlooking.

2.1 GENERAL SYSTEMS THEORY

A system is a set, or group, of related elements existing in an environment and forming a whole. Systems can be made up of objects (computers), subjects (your employees) and concepts (language and communication); they can be made up of any one or more of these elements. There are 'real systems' (those which exist independent of an observer), and 'conceptual systems' (those which are symbolic constructs). Our system, 'The anti-virus strategy system', is not so different from many others, in that it is composed of all three elements: computers (objects), people (subjects) and concepts (policies and ideas). Each of these systems has its own subsystems. For example, your system of networked computers consists of individual computers. These computers are comprised of yet more subsystems; microprocessors, resistors, disk drives, etc. Our system consists of both real and conceptual subsystems. A system can also be said to be a way of looking at the world, or a point of view [5].

Concepts, laws, and models often appear in widely different fields [6] based upon totally different facts. This appears to be at least in part due to problems of organization, phenomena which cannot be resolved into local events, and dynamic interactions manifested in the difference of behaviour of parts when isolated or in higher configurations. The result is, of course, a system which is not understandable by investigating their respective parts in isolation. One reason these identical principles have been discovered in entirely different fields is because people are unaware of what those in other disciplines are doing. General Systems theory attempts to avoid this overlap in research efforts.

There are two main methodologies of General Systems research; the empirico-intuitive and the deductive theory. The first is empirical, drawing upon the things which regularly exist in a set of systems. It can be illustrated fairly easily, but lacks mathematical precision and can appear to the 'scientist' to be naïve. However, the main principles which have been offered by this method include differentiation, competition, closed and open systems, and wholeness – hardly naïve or worthless principles. The second method, basically, can be described as 'the machine with input', defined by a set 'S' of internal states, a set 'I' of input and a mapping 'f' of the product I x S into S (organisation is defined by specifying states and conditions). Self-organising systems (those progressing from lower to higher states of complexity, as in many social organisations) are not well suited to this approach, as their change comes from an outside agent. Our anti-virus strategy system is such a system and for this reason we will use the empirico-intuitive methodology.

Classical system theory uses classical mathematics to define principles which apply to systems in general or to subclasses. General System theory can be called the doctrine of principles applying to defined classes of systems. It is our hope that we can stimulate thought on how already-known principles can help us in managing our anti-virus protection by examining the system as a whole.

2.2 HOLISM

Our definition of holism, drawing where appropriate from the medical profession, is health-oriented, and focuses on maintaining and improving the existing health of the system. It does not focus on disease and illness. It is interesting to note that, while we have many terms that relate to compromised and infected systems, we do not seem to have many terms relating to 'well' computers. Holism operates under the assumption that the open system possesses an innate organising principle, with the interdependence of the parts having an effect on the total system health. Holism views symptoms of distress as signalling disharmonic conditions, from which we can learn how to adjust the system (feedback); it is open to a variety of approaches for attaining balance. The focus of holism is heavily slanted toward the correction of causal factors, not symptomatic relief. Thus, the role of the holistic practitioner is to facilitate the potential for healing [7].

3 ANTI-VIRUS STRATEGY SYSTEMS

Where do our anti-virus strategy systems fit in this picture? We hope to explore some answers to that question by first examining the components of our model system. Keep in mind, however, that the goal of this paper is not to provide you with answers, but rather to stimulate new ways of thinking about the problems we face daily.

3.1 COMPONENTS

Each of the components in Diagram 1 contributes to the overall health of the system. Conversely, each can contribute to the illness of the system. For instance, our computer can contribute to the health of the system by functioning properly. If the hard drive crashes, a disharmonic condition is introduced. Our managers contribute to the overall well-being of the system, as long as they perform correctly. However, if one of them intentionally or unintentionally infects a computer with a virus, he or she contributes to the illness of the system. Our software contributes to the wellness by keeping employees reassured, and by keeping viruses out. If it is disabled by an employee desirous of more speed upon boot, or if it does not do its job in virus detection, it contributes to the illness or chaos in the system. There are other factors not shown, as the anti-virus strategy system model does not stop at the boundary of the company. The model includes your Internet service provider, virus writers, makers of electronic mail front-ends, anti-virus product tech support people and more. For the purposes of this paper, we must draw an artificial boundary. We mention the rest to give you food for thought, and to illustrate that boundaries are not static.

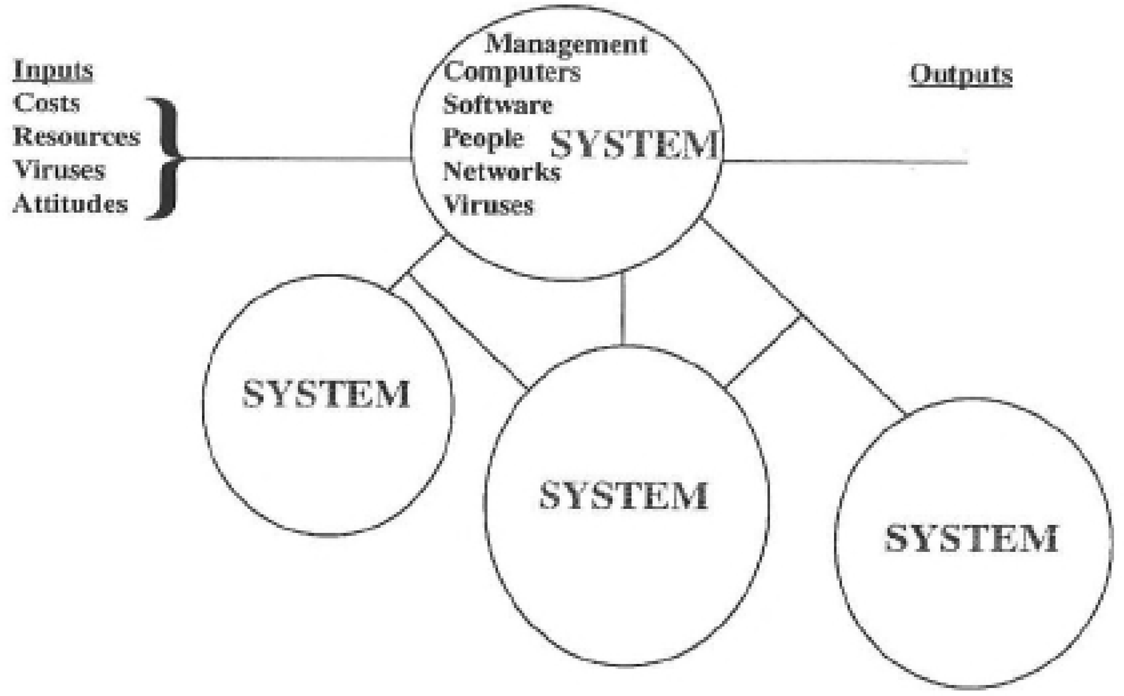

Figure 1. Anti-virus Strategy System - The Environment

3.2 PROGRAMS POLICY AND PROCEDURES

(SELECTION, IMPLEMENTATION AND MAINTENANCE)

Where do we begin in examining the interaction of our chosen system elements? Let's start with the software selection. Anti-virus software is selected based on a wide number of criteria (8). While some of these criteria are beneficial, several are counterproductive at best (9). We need to be aware of exactly how our company's software is being chosen, and not leave this vital aspect of software selection up to people who do not have the experience or expertise to make a selection that will maximize your organisation's protection against viruses.

Does your anti-virus software detect all of the viruses which are a real threat to your organisation? Before you glibly answer yes, you should recognise that all products are far from created equal, and that even the best products will not achieve this goal if not properly maintained. Consider the following:

'When asked what happens to two blocks of copper initially at different temperatures left alone together in an insulated container, students will reply that the blocks will come to the same temperature. Of course, if asked how they know, they usually say ''Because it is a law of nature''...the opposite is true...it is a law of nature because it happens. [10].'

Apply this to your anti-virus software. Does it catch viruses because it is anti-virus software? If so, you can depend on it, as its name defines what it is. But, if you even loosely apply this concept, you will see that it is anti-virus software because it catches viruses – and if it does not, then what does that make it?

Remember the following quote:

'If you call a tail a leg, how many legs has a dog?'

'Five?'

'No, Four. Calling a tail a leg doesn't make it a leg' [11]

Maintenance of your software is another critical issue. Maintenance refers not to the upgrade, but to the maintaining of the software on a daily basis. What does it require to run? Are you supplying what it needs to live? Or is it merely surviving? Does it have adequate memory, power, disk space to run optimally and lessen the chance your employees will disable it? Is it in an environment free from other programs which may hinder its performance? If you cannot answer yes to these questions, you are not providing an environment for this element of your strategy system which will allow it to remain viable. It will not survive. Like living systems, the anti-virus strategy system requires a favorable environment, else the system will adapt. Unfortunately, in the case of this system, adaptation can mean software becoming disabled by the user component of the system, or overridden by a competing software component. All this, and we have not even added viruses which by design cause a problem to the system by the introduction of instability.

Even if you have the best anti-virus software, and are running it optimally, there can still be problems. Software is just one part of the strategy system. Policies and procedures play an important role in the overall strategy. Even the viruses we mentioned earlier play a part in this system. Then there are the least predictable aspects of the system, the human beings. How complex is this system? How much should we expect the people involved to understand?

Ackoff defines an abstract system as one in which all of the elements are concepts, whereas a concrete system is one in which at least two of the elements are objects [12]. As you can see, our system is concrete. It is also by design an open system, one into which new components may be introduced. Some of these components are by nature 'unknown' (i.e. actions of people, how software may react, viruses which may appear).

When these components are introduced, we have to consider first how they behave on their own. Next, we have to consider how they would behave in combination with any and/or all of the other elements. Finally, we have to consider how 'things' in general will be if neither of the objects are present. In its most simple form, a two-part system would require four equations, but of course, you can see that as the number of elements increases, the number of interactive equations grows by leaps and bounds [Table 1].

	Linear Equations			Nonlinear Equations		
Equation	One Equation	Several Equations	Many Equations	One Equation	Several Equations	Many Equations
Algebraic	Trivial	Easy	Essentially impossible	Very difficult	Very difficult	Impossible
Ordinary differential	Easy	Difficult	Essentially impossible	Very difficult	Impossible	Impossible
Partial differential	Difficult	Essentially impossible	Impossible	Impossible	Impossible	Impossible

Table 1. [From [5]] - Introduction of Elements

One of the systems theory approaches we can draw from here to help illustrate the problem comes from what is sometimes called the Square Law of Computation. This means basically that unless you can introduce some simplifications, the amount of computation involved in figuring something out will increase at least as fast as the square of the number of equations. Consider all of the interactions between humans, computers, and software, and you will see why it is impossible to precisely calculate what the results of all of those interactions will be. We cannot even measure them. In other words, you cannot possibly anticipate all of the problems you will encounter in trying to keep your company's data safe from viruses, because you cannot possibly calculate the interactions which will occur once you begin trying to formulate a strategy. Needless to say, these interactions create 'problems'.

If we examine our anti-virus strategy in various ways, we may be able to see things more clearly. Another helpful way in which we can view our system is as an expression, such as the terms of a set. For instance, the notation:

Let x stand for marriage

Let y stand for carriage

Let z stand for bicycle

The set [x,y,z] is simple enough for anyone to understand. Using names in sets takes us to the more complex:

[The look on your face when you saw your first child, a proof that Vesselin Bontchev is not the Dark Avenger, an atom of plutonium]; wherein the first no longer exists (or possibly never did); the second has not yet existed, and the third is out of reach of the common man.

If you were to be asked for the meaning of the ... in the set [Alan, Dmitry, Fridrik...] would you say the ... represented men's names? Names of programmers? Names of programmers who make anti-virus software? Names of people not from the United States?

What is the rule for determining the meaning of what is unstated? Is there some unwritten heuristic of which your employees are not aware? What is the meaning of the three dots in our set?

This has a particular application to policy. Users can easily understand, 'Do not turn the computer off if you find a virus'. Can they as easily understand, 'Do not reset the computer if you find a virus'? Can they understand, 'In the event of a suspected virus, call the administrator or take appropriate action'? What *is* a suspected virus? Is it any time the computer system seems to act strangely? Is it only when the letters fall off? After all, that's what viruses do, right? What is appropriate action? [Turn off the computer, Call your supervisor, Reboot the computer, ...] What is the meaning of the ... in this set?

4 VARIATIONS ON A THEME

How well are our strategies doing? As pointed out early on, not very well. Why not? To help answer that question, next we will examine the problems of our strategy using the concept of variation. We recognise the duality of variables as they relate to information processing; the significant values which variables acquire at the two extremes of their respective spectra. Specifically, in order for a system to continue to thrive, information must be processed. Disorder, uncertainty, variety – all must shift from high to low [Table 2].

Disorder, Uncertainty and Variety: Entropy and the Amount of Information Processed

High	Disorder	Low
High	Uncertainty	Low
High	Variety	Low
Large	Number of Alternatives	Small
Small	Probability of an Event	Large
Low	Regulation and Control	High

Table 2 - Predictable Output

The probability of particular events follows by decreasing from small to large. The amount of regulation and control increases from low to high. We become increasingly sure of the output of our systems [13]. However, viruses introduce a form of disorder with which the human components of our systems are not intimately familiar. While the probability of infection can be calculated mathematically [14], we are unable to calculate the probability of other events related to viral infections[15]. In what ways does this introduced unfamiliarity manifest itself? One manifestation is the appearance of problems.

We typically try to solve most of these problems deductively, to determine the reason for a variation between design and operation or design and implementation. This approach is doomed to failure because it places the blame on the subsystems. We attempt to 'restore to normal' instead of redesigning our system. We formulate plans based on incorrect, incomplete or obsolete assumptions. We neglect to factor in spillover effect, that is, the unwanted effect which actions in one system can have in another. Improving an isolated system may seem the epitome of system integrity. You can have your pure clean computer. Of course, it is virtually useless, unconnected to the rest of the world. Or, perhaps it is the solution. Isolated perfect machines. This would probably create a dissatisfied workforce, however, which would ultimately impact business negatively. In the case of anti-virus strategy, 'spillover' takes on many new dimensions – as many as the human beings with which our machines interface. Can you control all of the aspects of this system? You cannot.

Another factor to consider is the size and extent of our system. Further insight may be gained by considering what is sometimes referred to as the generalised thermodynamic law, which states that the probable state is more likely to be observed than the less probable. While this may incite the physicists among us, it has two parts which correspond to the first and second law of thermodynamics. The first law is hardly worth mentioning (physical reason), but the second is of interest to us. We should be concerned with the limited power of observers when viewing large systems. In other words, we cannot expect our managers to be in every place at once, knowing what is going on with every system, every employee. The concept of boundaries can be used to help solve this problem, but their definition is beyond the scope of this paper [16].

4.1 SYSTEM FAILURE AND MEASUREMENT

We say the system is failing for three reasons. It is not performing as intended. It is producing results other than expected. It is not meeting its goal. The objective is **NO VIRUSES.** However, in addition to often neglecting to define what 'no viruses' actually means, we are frequently unaware of how 'no viruses' can mean different things to different people. Not performing as intended could mean it finds some viruses but not all, or it finds all but only removes some. Unexpected results could mean it crashes 1 out of every 6000 machines, or produces system degradation you did not anticipate (if this is the case, does the fault really lie with the product for producing the degradation or you for not anticipating?) Not meeting its goal most likely means failing to keep out viruses. However, to some people, this is a different goal from 'no viruses'.

How is this possible? Isn't 'no viruses' a simple concept? In a word, no. When there is a malfunction, i.e. a virus is found, the natural tendency is to look for the cause within the system. We tend to blame the problem on the variation of the system from its 'desired' behaviour. It could be the fault of the program, the employee, the policy. We tend to blame the program as it is the part of the system most closely identified with the failure as immediately perceived. However, consider for a moment that, to your employee, 'no viruses' means simply that. No viruses are found. Following that line of thought, finding 'no viruses' would be a system success – that is, until it brought your operation to a halt. You see, to some people, 'no viruses' means that none are seen or observed, and not that none are actually operational in the system. We plan grandiose policies and procedures around finding a virus and make no space for 'no viruses' as a possible failed variation. If you find 'no virus', you need to be very sure it is not due to your employees disabling your software, or your software not finding the virus.

Many system 'improvements' are possible which in reality doom the system. Faulty assumptions and goals are often at the root of this problem. For instance, it is obvious that all of your computer workers must, under dire penalty, refrain from bringing disks from home into your office. You implement this policy. You assume they will comply. Your goal is compliance, not 'no viruses'. If the goal was 'no viruses', you would be forced to be more realistic.

Consider the following two statements:

'We have clean, working computers and by not bringing in software, we can keep them that way. It will save us all a lot of time, and effort!'

'If you bring in disks, you will probably infect our office computers. It will cost us all a lot of money.'

In the first instance, the focus is on the well machine. Everyone wants well machines. People like to be part of winning teams, and participate in things that are nice.

In the second, the focus is on the sick machine. None of your people would have viruses on their home computers. So, this must not apply to them. And if they do break the rule, you have already set them up to be afraid to tell you. After all, they don't want to cost you a lot of money and they certainly don't want to be known as the culprit for infecting the office computers.

How do we measure the performance of our anti-virus strategy system? Not very well. If we find some viruses, we say it's working. If we don't find any viruses, we say it's working. In some cases, you can apply 'we say it's not working' to these same sentences. There is no standard way in which we measure the success of the entire system. Only in the act of being out of control will the system be able to detect and bring back the control.

5 CONCLUSION

The systems approach proposed here is a 'whole system' optimization. Think of it as the configuration of a system which will facilitate optimal performance. There exists, of course, a dilemma, in that at some time suboptimization may be necessary, or even the only possible approach. An approximation which is used may be a great deal better than an exact solution which is not [17]. Nevertheless, our model will attempt to show ways to optimize system performance.

Models are how we express things we want to understand and possibly change, designed in terms of something we think we already understand. Models sometimes present problems when you try to translate them into real world activities. With this in mind, I would like to suggest a simple model which may help us begin to find ways to find a solution to the problem of designing a workable anti-virus strategy.

'Models should not so much explain and predict as to polarize thinking and pose sharp questions.' [18]

Using a holistically modelled approach, we would strive to maintain the existing health of the system. This assumes we have a healthy system to begin with. This requires you not depend on your belief that your software is correctly installed and operational, and that your employees know how to use it and are using it, and that your equipment is functional, and that your policies are correct and being followed... It requires that you actually take it upon yourselves to designate people to ensure that your system is optimal to begin with. If you are not willing to do this, you cannot expect to restore the system to health. The focus should shift from 'blame' to 'responsibility'. This may require investment on your part. You may need to update equipment. You may need to train employees. You may need to purchase software. You may need to subscribe to publications which can keep your employees up to date on trends in virus and security matters.

You will need to monitor feedback between various aspects of your anti-virus strategy system. We have not discussed feedback at any great length in this paper, due to the number of elements of the system and the complexity of the feedback. However, using the empirico-intuitive General Systems theoretical approach defined earlier in this paper, you should be able to determine the sorts of feedback which are required to keep your system functioning optimally. If there is NO feedback, you can rest assured your system will fail. Lack of feedback produces entropy. In simple terms, entropy can be called the steady degradation or disorganization of a society or a system. This is not what you want for your system. You want to move the system into organisation and order, high rates of probability and certainty. As we discussed earlier, this happens when information is processed. The information can be communication of any type between any elements of the system.

Our current focus seems to be on the existing illnesses in our systems. If open systems indeed, as suggested, possess an innate organising principle, perhaps we should be paying more attention to what the elements of our systems are telling us. We could learn the sorts of information required to maintain organised reliability. We could learn the amount and types of feedback required to process information optimally, and to keep the system both desirably adaptive and from adapting negatively. We must examine our systems as a whole, including all of the parts, as best we can, to determine what the elements and the system are telling us. In the case of our anti-virus strategy systems, we have yet to determine what that message is. Many of us have not even yet defined the elements of the system, the system boundaries, or the goal of the system.

It is clear that there are disharmonic conditions in the 'Anti-virus strategy systems' of most companies; if there were not, no one would be attending this conference or reading this paper. It is also clear that the way we traditionally approach these problems is not working. We have been using these approaches for a long time, and the problems are not going away. Drawing from the holism model, one thing we can do is examine causal factors, instead of focusing on symptomatic relief. We need to examine more closely the interdependence of the parts of our system, and as security professionals, should facilitate the potential for healing our systems. It is hoped that some of the ideas mentioned in this paper can provide a starting point for this.

The author would like to thank Louise Yngstrom, University of Stockholm, for late night chats on System Theory, above and beyond the call of even academic duty.

BIBLIOGRAPHY

[1] 'Virus Encounters, 1995: Cost to the World Population'. Testimony, House Subcommittee on Telecommunications and Finance, Tippett, Peter, June 1993.

[2] 'The Wildlist'. Maintained by Joe Wells.

[3] 'Real World Anti-Virus Product Reviews and Evaluation'. Gordon, Sarah and Ford, Richard, *Proceedings of Security on the I-Way*, NCSA, 1995.

[4] 'An Introduction to General Systems Thinking', p.3, Weinberg, Gerald. John Wiley and Sons, 1975.

[5] 'An Introduction to General Systems Thinking', p.51, Weinberg, Gerald. John Wiley and Sons, 1975.

[6] 'General Systems Theory: Foundations, Development, Applications', pp.xix-xx, Revised Edition, von Bertalanffy, Ludwig. George Braziller, Inc, 1980.

[7] 'Health Promotion Throughout the Lifespan', Edelman, Carole and Mandle, Carole. Mosby, 1994.

[8] 'Guide to the Selection of Anti-Virus Tools and Techniques'. Polk, T. and Bassham, L. NIST Special Publication 800-5. NIST, December, 1992.

[9] 'Real World Anti-Virus Product Reviews and Evaluation', Gordon, Sarah and Ford, Richard. *Proceedings of Security on the I-Way*. NCSA, 1995.

[10] 'Semantics, Operationalism and the Molecular-Statistical Model in Thermodynamics', Dixon, John and Emery, Alden. American Scientist, 53, 1965.

[11] Quote attributed to Abraham Lincoln.

[12] 'Applied General Systems Theory', p.39, Van Gigch. John P. Harper and Row, 1974.

[13] 'Applied General Systems Theory', Figure 2.2, Van Gigch. John P. Harper and Row, 1974.

[14] 'Directed Graph Epidemiological Models of Computer Viruses', Kephart, Jeffrey O. and White, Steve, R., *Proceedings of IEEE Computer Society Symposium on Research in Security and Privacy*, 1991.

[15] 'The Viability and Cost Effectiveness of an 'In the Wild' virus scanner in a Corporate Environment', Gordon, Sarah, 1995.

[16] 'Applied General Systems Theory', p.25, Van Gigch. John P. Harper and Row, 1974.

[17] 'The Development of Operations Research as a Science', pp.59-60, as cited in [4]. Ackoff, Russell. *Scientific Decision Making in Business*.

[18] 'Some Mathematical Models in Science', Kac, Mark. Science, 166 No. 3906 695, 1969.

BLESSINGS IN DISGUISE: BUILDING OUT OF DISASTER

Paul Ducklin

Sophos Plc, 21 The Quadrant, Abingdon, OX14 3YS, UK
Tel +44 1235 544037 · Fax +44 1235 559935 · Email duck@sophos.com

Imagine, for a moment, that there is such a thing as the 'average anti-virus expert'. Take him or her aside briefly, and start talking gently and generally about computer viruses. The chances are good, even if you are skilled in keeping a conversation running along lines of your choice, that the subject matter will veer rapidly towards the technical. You should not be surprised if the expert makes a sudden subroutine call to highly technical matters; you should be even less surprised if you find that the subroutine stack becomes lost, so that a return to the original topic of discussion is impossible.

Likewise, much of the literature published in the short lifetime of the anti-virus field is largely technical in content. Even documents which are supposed to be corporate anti-virus policies, written in 'plainspeak' and signed by Topmost Management, sometimes manage to lose themselves in a tangle of 'technobabble'. This can be hard to avoid if you are trying to describe the best way to navigate through a nightmare world infested with armoured, tunnelling, full-stealth, highly polymorphic, multi-partite, fast-infecting malware objects.

Sometimes, though, the problems which emerge from the technological computer virus battleground are decidedly plain. Often, the 'obvious' attack (the unsubtle, widely telegraphed, low-tech viral broadside) succeeds where deviously clever programming fails. It may be an old bromide [1], but we can hardly blame technology for the ongoing prevalence of viruses such as Form and Stoned [2]. In July 1994, another technically unremarkable virus succeeded globally, making a sudden appearance worldwide: Kaos4.

This paper is a case study of an attack by this virus on a large South African company. Managers, network administrators, and users alike were all surprised by the sudden appearance of the virus; they were even more surprised when the virus reappeared just as suddenly three months later. As you will probably guess, these were not innocent, defenceless victims – especially as the lessons learned after the first attack should at best have prevented the second, and at worst allowed it to be handled with ease.

However, as an internal survey has shown, this organisation's corporate anti-virus awareness has improved as a direct result of the Kaos4 incidents, and the risk of viral disaster in the future has been addressed, and reduced. It would probably be going too far to describe Kaos4 as a blessing in disguise for this company (and it would offer the virus itself a function and legitimacy it does not possess), but they have certainly managed to learn from their mistakes. As will become obvious, though, they did not learn quite as swiftly as they might have done; others will want to aim to do better, faster.

To maintain the privacy of the company studied here, we shall refer to them as 'The Company We're Studying, Limited', and abbreviate this name as 'TCWS'. And, before you smirk at their story, ask yourself if you are absolutely certain that it couldn't happen to you.

THE INTERNET SPEAKS

According to the Internet, this is what happened (messages have been edited to remove personal or commercial identifying information; errors are reproduced as they originally appeared):

```
Date:    Thu, 28 Jul 94 07:58:11 -0400
From:    ABC Anti-Virus Company <abc@def.ghi>
Subject: Virus warning (PC)

We have discovered an infected file which has been spread on Usenet
in the group 'alt.binaries.pictures.erotica'. The virus is called
Chaos4/ kohntark 697, and is a com/exe infector. No current scanners
seem to be able to detect it yet. A detector/disinfector routine is
available in the file 'abcdefg.zip', which has been uploaded to
several sites (wuarchive.wustl.edu, ftp.funet.fi,
ftp.informatik.uni-hamburg.de etc.)

Sincerely,
S. O. Meone
ABC Anti-Virus Company

_ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _ _

Date:    Tue, 30 Aug 94 01:04:37 +0400
From:    XYZ <xyz@uvw.rst>
Subject: Re; [News] KAOS? (PC)

Hi !

somebody@somewhere.com (Some Body) writes:

> I have been hearing about a new (?) virus called KAOS that
> has been transferred over the internet. Does any one have
> any info on it?

Any name of this virus is 'Kaos4'.

So far, verified reports or samples of this virus have been received
from the US, Austria, Norway and Finland.

It seems that the virus was distributed over Usenet, possibly in one
of the alt. groups.

The virus is not very remarkable - it is a 697 byte non-resident
COM/EXE infector, which contains the string 'KODE4 / Kohntark' (The
'o' has 2 dots above it). This string is not encrypted and can be
```

```
found with any text search utility.
The virus does not seem to have any specially interesting functions,
and does not contain any destructive code, so the problem is not as
serious as it might have been, but the virus might have non-
intentional side-effects, such as preventing a machine from booting
if it infects IBMBIO.COM/IBMDOS.COM on a machine running IBM DOS.

—XYZ

- - - - - - - - - - - - - - - -

Date:     Sat, 01 Oct 94 11:20:46 -0400
From:     Another Person <ap@pqr.stu>
Subject:  Re: Re; [News] KAOS? (PC)

XYZ (xyz@uvw.rst) wrote:

> Any name of this virus is 'Kaos4'.
> So far, verified reports or samples of this virus have
> been received from the US, Austria, Norway and Finland.

and South Africa...

Cheers, Another Person
```

Ignore for the moment the obvious inconsistencies in the above messages; the bulk of the information explains what happened, and how the virus came to make momentary global headlines. From a viral point of view, Kaos4 was interesting because of a novel combination of factors:

- the virus was uploaded openly to the Internet, posted into a newsgroup, which acted as a vehicle to spread the virus rapidly across the globe
- the uploaded infected object was a commercial shareware program
- this program was a game
- the upload destination on the Internet was an unmoderated newsgroup better known for disseminating pornographic pictures
- the uploader made no attempt to conceal himself, acknowledged the upload and claimed that he had taken appropriate precautions, but had been let down by an anti-virus scanner which gave his upload a false bill of health.

MYTHS CONSOLIDATED

The nature of the Kaos4 distribution gave the virus multiple angles of importance in the media, and in tearooms across the corporate world. Unfortunately, the immediate lessons learned from it, judging from informal conversations with victims at TCWS during my consultancy immediately following their first attack, were not particularly useful. In some cases, they served only to perpetuate those inaccurate myths which arose years ago, when the computer virus first began to become a problem [3].

Computer games have long been singled out as 'dangerous' software, and many companies have banned games because of the viral risk they pose – rather than because their employees are usually taken on to perform tasks other than playing games at work. In fact, games may even serve a valuable purpose in business (indeed, I learned to type fast playing the curiously-named 'Lobster Sea Adventure', designed specifically to teach keyboard speed and accuracy). Worse still, suggesting that there is a specific type of 'dangerous' computer program serves to suggest that there are programs which are inherently 'safe', and therefore immune to viral infection. Since many file viruses will infect just about any executable file which comes their way, this is an unsafe myth.

Many companies also retain shareware and freeware on their list of 'dangerous' programs. Clearly, it is the means by which shareware is sometimes distributed that needs consideration, rather than the shareware itself. After all, a careless or unscrupulous dealer selling shrink-wrapped commercial software might easily use, infect, and re-package that software (and such cases are well-documented) – so singling out the shareware concept as virally risky simply serves to deflect attention from the facts.

The burying of Kaos4 in a pornographic newsgroup also obscured corporate risks, with managers and users who steered clear of such newsgroups tending to believe that this would keep them and their workgroups safe from the virus – in the same way, perhaps, that those who are not sexually promiscuous reduce their risk of HIV infection to negligible levels.

Unfortunately, the obvious analogy here is not the correct one; a more useful analogy would be the observation that because you and I are not thieves does not mean that our houses are, *ipso facto*, safe from burglars. Furthermore, once a virus is in the wild, its initial mode of distribution becomes relatively unimportant – once Kaos4 was out there, companies not even on the Internet were at risk from infected floppies, just as they would be with any other virus.

The fact that Kaos4 was connected with not one but **four** arenas often proscribed to corporate users – games, shareware, bulletin boards (or the Internet, seen by some as a giant global BBS) and pornography – has led some observers to conclude that the uploader of the virus deliberately sought out a multiply-prohibited way in which to introduce the virus. This, presumably, would delay initial reports of the virus inside an organisation, with users possibly quadruply afraid of recriminations. You can imagine that the average user, complaining to Technical Support about a new-found problem with his or her PC, would be unlikely to offer a report such as, 'I downloaded a shareware porno game file called SEXY.EXE from the Internet, installed it on my work PC and ran it, whereupon the network fell apart'.

SOMEONE ELSE'S FAULT

Although access to the Internet is increasingly popular, and increasingly important to business, it is still often seen as an unconquerably dangerous vehicle, and incidents such as Kaos4 serve to fuel fear and ignorance about the safety and useability of the Internet. Many corporate computer network and security policies deal with the Internet in simple, total fashion – interconnectivity with the 'Net' is prohibited. In the wake of Kaos4, the Internet was certainly a convenient hook on which to hang blame.

Within TCWS, who had (and have) partial Internet connectivity, one of the outcomes of Kaos4 was to concentrate security attention on the interconnection point. Whilst network firewalls do indeed require expert attention, TCWS's firewall was configured quite restrictively, and was already fairly carefully managed. Their internal networks, on the other hand, were not. If the organisation's internal policymakers had stepped back before the Kaos4 incident, they would doubtless have prevented the first attack with ease. If they had resisted the temptation to zoom in afterwards, they would have prevented the second attack.

BATTLE STATIONS

The first TCWS engagement appears to have unfolded like this:

- The actual infected game, as uploaded to the Internet, was brought into the company. Whether it was downloaded directly from the Internet or transferred to an employee by someone outside is uncertain, although it seems that the latter was the case.
- The game was tried on a handful of PCs on the corporate network, which all became infected.
- Infection spread from one of those PCs to one of the company's NetWare servers, and executable objects which were part of the corporate mail system were infected.
- Other servers on the network became infected when control workstations attached to both the mail server and one or more of the user file servers executed infected mail software.
- Workstations were rapidly hit as infected programs on infected servers were accessed by users.

None of the anti-virus scanners in use at TCWS detected Kaos4, which is unsurprising: presumably, the author went to some trouble to ensure that this would be the case, and relied on the virus travelling so far, so fast, that some victims would be hit before their scanner updates arrived.

Scanner detractors will be quick to point out that this is an unacceptable and ongoing weakness in scanner-based protection, and that integrity checkers would have noticed this viral attack as soon as it started. Thus, TCWS's problem was that they had elected to use the wrong anti-viral tools. Scanner versus integrity-checker arguments have become, in some quarters, the 'Holy War' of the anti-virus community; fortunately, there is little need for us to consider this debate here, because TCWS's problem was rather different, and much broader than this.

Firstly, although several brands of anti-virus software were in use within the company, TCWS had a corporate licence for one specific product. This licence permitted them to install the software on all user workstations, where the built-in integrity checker of the software would have identified the spread of the virus early in the day. Additionally, their vendor was consulted by telephone, and immediately provided a virus database update which allowed TCWS to detect Kaos4.

Some departments of TCWS swung into action quickly, and removed the virus from PCs in their corridors. Others had never got round to installing anti-virus software on their systems in the first place (though the company had bought and paid for corporate protection some time before). They were much slower to react, because they had never carried out a dry run – they left the anti-virus learning process until they were faced with the 'Real Thing', whereupon fear and panic added themselves to the equation and frustrated their attempts to do things correctly and efficiently.

One or two departments took the extreme approach of closing down all their PCs, disconnecting from the network, and laboriously cleaning every PC before going live again. Sadly, the fact that the corporate clean-up was neither co-operatively performed nor centrally run meant that they beat some of the more lax departments to it. Furthermore, they merely removed the virus, and paid little attention to their current network configuration. So, when they brought everything back up again, it was back to square one. Just as the virus had entered and propagated across their server and workstations before, so it did again.

THE ENEMY WITHOUT

From the scope of the attack, you might conjecture that Kaos4 is a 'difficult' virus, with tricks such as fast infection and stealth to help it spread far, fast, and unobserved. Actually, it is a very plain little virus. Kaos4 is a direct action (non-memory-resident) file infector; it has no stealth capabilities, making no effort to disguise itself; and it is entirely unencrypted, so that the text string 'KAOS4' (not 'KODE4' as stated in the Internet posting above) is clearly visible in every infected file. It is, in a word, obvious.

Because Kaos4 is a non-resident, non-stealth virus, it is the kind of file virus that a network administrator, faced with a compulsory viral infection, would be well advised to choose. Indeed, even the 'Golden Rule' of virus hunting – boot from a known, clean, write-protected system diskette – can be ignored in the face of Kaos4. Even without any anti-virus software, a network administrator would need very little to get his network back on its feet again: minimal programming skill; a BASIC compiler (or even a simple programmer's tool such as GREP); and a network operating system which supports login scripting would be enough to do the job.

It is interesting to note that, in my wanderings around TCWS whilst on contract to investigate the nature and extent of their problem, I came across a small workgroup who had decided to take on Kaos4 themselves. Although most computer scientists who have ever been involved in technical support might shudder at the thought of a band of 'Have-A-Go-Henries' within a community of computer users, their approach demonstrates that careful thinking, combined with the use of obvious precautions, can often produce a high-speed one-off solution to an apparently large problem.

They noticed that program files on their workstations were growing in size, and surmised that they had a virus. One of the team set about an immediate backup of the workgroup's PCs; two others began to examine the altered files. Although they knew very little about viral replication, they compared infected objects with fresh originals, and deduced enough about the relationship between the two to guess how to convert infected files into clean ones.

One of the two had done a little Pascal programming, and whipped up a utility to apply the conversion scheme they had deduced. A little testing, both of the hypothesis and their utility, and they were ready to try it out. By this time the backup was complete, so they had little to lose – instead of waiting for their network administrator to visit their machines personally, they were the first on the block to be up, running, and clean. In some companies, you could probably get sacked for that sort of behaviour – but these three 'Wild West' problem solvers got away with it. Their utility worked; they simply neglected to mention it to their administrator, and quietly allowed him to take credit for cleaning their workplace. And they did make a proper backup first, so their experiments were relatively risk-free.

THE ENEMY WITHIN

If someone tosses a lighted match through your letterbox, burning down the house, they would be guilty of arson. They ought to have foreseen that your house might burn down, and should be punished accordingly. At the same time, if you knew such an attack was likely, it would be a wise move to buy a fire extinguisher, and to learn to use it. It would also be prudent to give up the habit of storing uncovered buckets of aviation fuel inside your front door.

In the same way, the ultimate responsibility for the attack on TCWS lies with the author of the virus – Köhntark, as he seems to call himself (there was, at the time of the first appearance of the virus, an attempt to establish some sort of forensic link between the person who uploaded the virus to the Net, and the author, though no connection was ever proved).

In terms of self-protection, however, some of the TCWS internal networks and procedures were veritable buckets of petrol, and this served to help the attack succeed. Most of the mistakes are fairly obvious from the battle chronology listed above. Their details are as follows:

- Untrusted software – even if its origin appears more benign than an unmoderated Internet newsgroup – ought not to have been used directly on any machine on the main network. Untrusted software should be track-tested on an auxiliary system first.

- The central mail system executables, although write-protected on the server, became infected during a SUPERVISOR login from an infected workstation. Because super-user logins grant unlimited power, they ought never to be allowed except from secure, trusted workstations. Additionally, in this case, the super-user login was used for a mundane task out of habit, not out of necessity.
- The virus spread from the central mail server to other servers, thanks to inappropriate access control configurations on those other servers. Programs had been unnecessarily deposited in a world-writeable data directory due to a misunderstanding about the requirements of the software.
- Workstations for which anti-virus software had been ordered, received and paid for were unprotected. Even though the software would not have detected Kaos4 directly, its integrity checking component would have provided rapid indirect notification of infection, if it had only been installed.

These configurational and procedural errors were confirmed rapidly during my investigation, and steps were taken to educate the company's network administrators about what was wrong, and what they could do at once to reduce the risk of reinfection. To what I thought was the extreme credit of the IT managers at TCWS, I was never asked to present them with information that could be used in a witch-hunt. The brief of my work was simple: find out what went wrong this time; help set things up to prevent it happening again.

The wisdom of TCWS in resisting a knee-jerk reaction, such as insisting on finding someone to punish, cannot be understated. In this case, it was recognised that numerous mistakes had been made, and that these mistakes had worked together to leave the corporate network insecure. There had obviously been no deliberate sabotage attempt; instead, the virus attack was seen as a 'total corporate quality' failure. With this in mind, an in-house seminar, open to all staff and paid for out of a central corporate budget, was scheduled and duly held.

Sadly, there was a repeat attack of Kaos4 at TCWS about three months later. Initially, when I was contacted again to look into the circumstances of this attack and to assist in cleanup, I felt a sense of personal failure. It was not as though I had been contracted to help reduce the number of virus attacks by 7%, or some such nebulous score. There had been one attack, and the new target was zero, which left little margin for error.

REPEAT PERFORMANCE

Cleanup the second time around was straightforward, because most of those involved had previous experience with this very virus. Additionally, I observed a number of things that made me feel much less personally concerned about the repeat attack. Consultants are often despised as those who talk about solving problems because they are incapable of actually solving them; by the same token, their role, especially in large corporates, is usually defined to stop short of implementing any solutions they devise. Consultants usually do not need to say 'I told you so', because that is what they were employed to do in the first place – and this is how it was at TCWS.

The repeat infections would have been prevented if the simple changes recommended three months earlier had been carried out. This time, there was no Internet to blame, as the reinfection started completely internally. Amongst the things which had happened or not happened since my previous involvement, were:

- Reinfected networks on which the purchased anti-virus software (now fully Kaos4-aware) had still not been installed three months later, despite the protestations of users.
- Reinfected networks on which infection had again been spread by shared programs on the server that were world-writeable.

- On-going routine use of the SUPERVISOR account from arbitrary user workstations, due to its convenience.

Clearly, the lesson to be learned here is that where changes are required, they must be seen to have been carried out. Network configuration can be monitored with auditing tools; the output of such tools can be cross-checked. Basic informal examination of network security by users, as well as by trusted outsiders, is a simple task. In a multi-departmental company such as TCWS, with departmental networks, administrators can easily assist one another with basic security audits. The idea here is not to watch for malicious internal breaches of security (that is a separate issue), but to prevent easily-avoided lapses which could prove costly.

A YEAR ON

One of the things which TCWS pushed most strongly after the first Kaos4 incident was the education of end users. My own hope, as the person contracted to run the first major seminar, was that users would be keen to attend. Before the TCWS incident, my experience had been of corporate anti-virus seminars being restricted by top management, in order to keep both direct and indirect costs down. TCWS, on the other hand, made every effort to remove this barrier, providing central funds and encouraging all staff to attend. Their belief was that the cost of the seminar in lost working hours would easily be recovered in hours saved handling virus problems in the future.

The users proved relatively uninterested, despite considerable publicity given to the event via corporate e-mail and through network administrators. In the end, less than 5% of potential delegates attended, although preliminary estimates suggested a turnout of over 12% was likely. I should like to be able to say why attendance was so poor in order to help other organisations avoid similar disappointments; sadly, the reasons were never clear.

It was clear, however, from a survey carried out amongst TCWS users one year after the original Kaos4 attack, that users continue to consider in-house anti-virus seminars unimportant. Users were asked:

```
Rate the following in terms of their importance to virus protection
inside the organisation (use the digits 1 to 5, with 5 for 'most
important', down to 1 for 'least important').

( 4 ) Formal corporate anti-virus policy
( 5 ) Anti-virus software
( 3 ) Network administration and configuration
( 2 ) Seminars, information sharing and awareness campaigns
( 1 ) General attention to 'total corporate quality'.
```

Unsurprisingly, anti-virus software was overwhelmingly voted most important. Seminars and awareness campaigns, however, were rated second last, just above 'total corporate quality'. Clearly, TCWS users do not see the anti-virus issue as their management do: whilst users put a formal corporate anti-virus policy in second place, they seem relatively unconcerned about getting themselves into a position to understand how they might build this policy into their own computing regimen. Whatever TCWS users may think, I agree with their managers, and rate anti-virus protection as a total corporate quality issue.

Nevertheless, TCWS users have a healthy understanding of their own importance in the corporate anti-virus battle. They were presented with:

```
Rate these people or groups in order of their importance to
controlling viruses inside the organisation (use the digits 1 to 5,
with 5 for 'most important', down to 1 for 'least important').

( 1 ) Top management
( 5 ) Network administrators
( 3 ) Computer maintenance contractors
( 2 ) Everyone else
( 4 ) Me.
```

Although they chose to place the bulk of the responsibility on someone else (they picked network administrators as most important), they voted 'Me' into close second, which is a good sign. Top management were placed in a very distant last place. Strangely, however, despite the importance associated with network administrators, it was unclear what TCWS users thought these administrators would be doing in dealing with viruses:

```
Rate the following items or activities in order of the viral risk
they pose to the organisation (use the digits 1 to 5, with 5 for
'highest risk', down to 1 for 'lowest risk').

( 4 ) Exchange of disks with outside companies
( 2 ) Use of disks to move information between work and home
( 1 ) Incorrect network administration and configuration
( 3 ) Software taken from bulletin boards or the Internet
( 5 ) Illicit copying of software from other people.
```

Incorrect network administration and configuration was felt to pose the lowest risk to the organisation. Although the inextricability of the link between virus protection and network security was stressed at the TCWS user anti-virus seminar, it would seem that there were not enough users there to hear the message that was preached that day. Piracy was rated most risky, with exchange of disks with other companies voted into second place; the Internet got off more lightly than I had expected, rated in third place.

WHAT NEXT?

The organisational culture lessons from this case study are clear, and somewhat surprising: even though your users may recognise the significance of their role in keeping the organisation virus-free, they may yet have a certain reluctance to learn. Their virus awareness may improve, but not as much as you might wish it to:

```
How would you describe your computer virus awareness of a year ago?

[ 7%] Excellent. Understood the technical and organisational issues
[30%] Good - confident I knew enough to handle one if I got hit
[46%] Fair - heard of them, and had some idea of how they spread
[17%] Poor - heard of them, but they were 'someone else's problem'
[~0%] Zero - never even knew that viruses existed.

And how do you describe that awareness now?

[ 7%] Excellent. Understand the technical and organisational issues
```

```
[40%] Good - confident I know enough to handle one if I get hit
[45%] Fair - heard of them, and have some idea of how they spread
[ 8%] Poor - heard of them, but they're 'someone else's problem'
[ 0%] Zero - never even knew that viruses existed until right now.
```

Satisfactorily, 23% of respondents claimed their knowledge had increased over the last year; 3%, surprisingly, said their knowledge had gone down. Nevertheless, even after a year during which computer viruses received a high profile inside the organisation, more than half the respondents effectively rated their own knowledge as insufficient to deal with a virus should they get hit. And hit they were:

```
Did you get hit by the Kaos4 virus in the past year?

 [ 5%] Yes, more than once
 [21%] Yes, once only
 [74%] No.

Have you had a virus *other than Kaos4* in the last year?
 [ 2%] Yes - more than one
 [13%] Yes - one only
 [85%] No.
```

This is a high virus incidence rate, and the administrative lessons here are obvious, and not especially novel: run your networks properly; use your anti-virus software; and make sure that when network reconfiguration is necessary, that it actually gets carried out. At TCWS, the IT administrators managed to make the same mistake twice. Your goal, of course, will be to make no mistakes at all.

REFERENCES

[1] Ducklin, P: 'Anti-Virus Education: Have We Missed the Boat?'; *Proceedings of the 1994 Virus Bulletin Conference*, September 1994.

[2] Whalley, I (ed.): 'Virus Prevalence Table', *Virus Bulletin*, June 1995.

[3] Greenberg, R & Rosenberger R: 'Computer Virus Myths', October 1993.

[4] alt.comp.virus: Usenet newsgroup - various postings, various dates.

[5] comp.virus: Usenet newsgroup - various postings, various dates.

HUMAN DIMENSION OF COMPUTER VIRUSES

Jean Hitchings

ICL Institute of Information Technology, University of Nottingham, Nottingham, NG8 1HL, UK
Tel +44 115 951 3356 · Fax +44 115 951 3353 · E-mail jean.hitchings@nottingham.ac.uk

ABSTRACT

This research considers the human issues when designing an information system that is resistant to computer viruses. Most information security considers technical factors but often ignores human issues. This paper begins by looking at the development of systems analysis and the compares it with information security. This is followed by a summary of current literature which indicates the importance of human factors.

Finally, there is a case study of a large multinational organisation where a computer virus infected the computer systems. The situation is analysed in context to current literature and the developments which are occurring in systems analysis.

KEY WORDS

Computer virus, information security, Virtual Methodology, Soft systems analysis, Human issues and computing.

INTRODUCTION

It is possible to compare information security to general systems analysis as both involve analysing an information system in order to determine requirements followed by a design phase. While methods to implement information security have remained relatively static, the last decade has seen the traditional approach to systems analysis (also known as hard systems thinking) questioned as to its suitability to information systems. A major problem with the traditional method is that it ignores the human factor. Information systems are considered in the same light as machines, assuming that they behave logically and as instructed. People are the main component of any information system and it is generally understood that they are not totally logical.

The human issues are manyfold and include the objectives of personnel (which may conflict with those of the organisation); the cultures of the people involved; and attitudes which can be influenced negatively by low morale or positively by a good *esprit de corps*. It would not be feasible to expect users to cooperate in designing a system which is going to make them redundant or cause them to carry out a considerable number of extra tasks in their day to day work.

The new approach to systems analysis is called the soft systems methodology [1,2]. It intends to include human issues in the analysis and design phases. In addition, the new methodology considers organisational issues, such as policies. If management policy is to praise staff only on visible output then it is quite feasible that a junior will concentrate on jobs which will help him to obtain recognition and ultimately promotion. Other tasks which may be more important, such as security procedures, could be totally ignored. This was certainly a major contributing factor to a large multinational organisation where outsiders were able to hack into the networked system. There were no obvious results to show management if time was spent on learning and implementing network security. The hackers were able to roam freely through the system with super user privilege, because this was the default. There is obviously a need to reconsider our approach to information security in order to avoid such situations in the future.

Another factor that should not be ignored is the environment within which the system and organisation operate. The environment includes for example, customers, competitors, and legislation. Competitors can affect a company in many ways. An organisation may be forced to produce an extra product or service, because competitors have introduced one and it is proving popular. Customers can affect organisations by only buying a certain product causing others to be discontinued.

The human issue can have a powerful effect on organisations and at last its significance has been considered by systems designers. However, this important issue is being ignored by those implementing information security and the traditional approach is still being used. The information security designer should be even more concerned with human behaviour as after all it is people that commit crime not computers.

Most information security breaches are committed by employees who are opportunists, have seen an opening in procedures and have taken advantage of this [3]. Now that information technology has moved into the open office there is even more opportunity to tempt employees. Coupled with this is the fact that managers in general appear to be unaware that the main threat is from within. Such managers are concerned with procedures that prevent outsiders from entering their systems and are much more lax with internal procedures. This was the situation in the case study described below where a virus infected an organisation's computers.

CURRENT LITERATURE

For some time it has been suggested that information security is not just a technology problem, but that it also concerns people. Davis and Price [4] state that security is a people related issue and give a number of reasons. Firstly, the system is designed by people and the original controls are dependent on their ability to understand the problems and the relevant solutions. The integrity of the system is also dependent on the people who build the system as well as the people who undertake the day to day maintenance. Once operational, the system is reliant on people to run it. They must carry out the security related procedures adequately, if the system is to remain secure. Finally, there is always someone whose level of control of the system is high. This person is necessary in most systems. They may be a senior manager with the authority to transfer large sums of money or perhaps a systems administrator who potentially has the ability to access any data or programs in the system and is responsible for the allocation of passwords to authorised users.

Morrie Gasser [5], devotes a section in his book to 'The Problem is People, Not Computers'. In this he explains that computer crime is usually concerned with breakdowns in procedural or personnel controls, rather than exploiting a weakness in internal controls. He concludes that so long as relatively easy non-technical methods exist to commit a crime, technical controls can largely be regarded as superfluous.

Often the case is put forward that computer systems are not particularly useful for detecting or preventing computer related crime because the perpetrators are usually employees that do not violate internal controls. Instead they tend to misuse the information or privileges which they are authorised to access or use. However, Gasser continues that, on reflection, it is often the case that people gain access to more information than they need. This may be because the security restraints have not been implemented or that it has been too costly or inconvenient to include them.

Wong and Watt [3]) devote a large chapter to people, 'People - Asset or Liability'. In this, they state that many cases of sabotage to computer equipment, data and systems are committed by employees. Also, most computer related fraud has been undertaken by trusted staff in organisations, sometimes colluding with outsiders.

A number of cases are described to support their claim, followed by a section on fraud prevention or reduction measures. These include traditional controls, such as, separation of duties, job rotation, and split knowledge or dual controls. Controls on inputs, outputs and amendments are discussed, along with structured walk-throughs of the system design and good documentation. Finally, there is a section on password management.

It can be seen that although the people aspect of information security has been discussed as a problem by several authors, it does not appear to receive the same attention in industry. This means that either information security personnel in industry are not aware of the issue, (which seems unlikely) or they consider the issue to be too difficult or too costly to implement.

The answer may be as indicated by Gasser, who gives the explanation that while it is relatively easy to detect a single bug in a system which can be exploited for individual gain or to the detriment of the organisation, it is much more difficult to totally eradicate bugs from the system.

THE ORGANISATIONAL DIMENSION

At the time of writing, only two publications have been identified which attempt to tackle security issues using a soft systems approach. Richard Baskerville [6] states that discussion of information security is restricted by the narrow influence of technology as the only solution. He feels this has prevented the field from expanding and keeping pace with developments within the area of computing.

Baskerville looks at the design of information systems security in the light of modern system analysis and design. He states that by discarding traditional information security approaches, it is possible to consider security as a variation of normal information system design.

Hitchings [7] develops an information security methodology called the Virtual Methodology (VM), which considers organisational, contextual and human issues as well as offering technical solutions.

It is clear that there is a genuine problem and current literature indicates that information security is a management issue which involves people. Over the last decade computing has developed rapidly, with advances in technology and in methodologies for systems analysis and design. In the area of information security great progress has been made in the technology being used, but there have been no real advances in the management of these techniques and the understanding of the role of human factors.

The methods for implementing security are outdated and a new methodology is required that takes into account the people problem. This methodology should ideally follow the soft systems approach, in keeping with current trends in systems analysis. By considering the human issue, an organisation may be better equipped to tackle the problem of information security and deter the introduction of computer viruses.

There follows a description of a case study where a virus was introduced into the organisation's computer system by an application programmer. It was allowed to happen because of organisational policy and carelessness on behalf of the employee.

THE CASE STUDY

1) Introduction

This case study refers to the accidental virus infection of a personal computer caused by unauthorised use of software. The personal computer was one of many in use in a large confectionery organisation providing the possibility of massive cross infection.

2) The organisation

The organisation is a major international confectionery manufacturer based in the UK with a significant world presence. Their strength in this market has been increased by a merger with another large company that also manufactures confectionery.

At the time of the incident the organisation employed more than 10,000 people in Europe and the UK with a turnover of around £1 billion.

3) Systems description

Information technology plays a major role in company business. Computer based systems cover all the commercial areas of the business, process control manufacturing, environmental controls, research and site security.

4) Technical details

The organisation is a medium to large IBM site and has a DEC based distributed network. Personal computers are used extensively throughout the company. Both physical and logical security access controls are employed.

5) Context description

The departments involved in the incident were the Finance Department of one of the factory sites, remote from head office, their local Information Technology Department and the Central Information Technology Department at head office.

The Finance Department was responsible to the factory Chief Accountant, who in turn was responsible to the Factory General Manager, who was responsible to the company Managing Director.

The personnel involved were the local Information Technology Coordinator, a local Finance Department Section Head and a Central Information Technology Analyst/Programmer. The Analyst/Programmer was temporarily responsible to the local Finance Manager and his role was to advise and develop a local specialist accounting system based on personal computers.

The local Information Technology Coordinator was also the local Computer Security Administrator and was therefore responsible not only to the Chief Accountant but also to the Information Technology Security Manager.

6) Details of the security lapse

The security breach involved the implanting of a computer virus into the personal computer that was being used to develop an accounting application.

The responsibility for personal computer security rests with the local departmental managers and the local Computer Security Administrator. Standards and guidelines for the use of personal computers in terms of physical, data and software security had been distributed to all personal computer users. However, there were no software controls in place to prevent or detect viruses.

7) Discovery of the problem

The Information Technology Security Manager at Head Office received a report from the Information Technology Manager at one of the other factory sites saying that unusual characters kept appearing on one of the personal computer screens. Following discussions, it was discerned that the personal computer had most probably been infected by a 'friendly' virus.

The personal computer was dispatched to head office and the origin of the virus traced as far as possible. It was decided that the source had been at one of the other factory sites.

At this point the incident was reported to Senior Management and it was agreed that the Information Technology Security Manager should investigate further and report his findings.

8) Organisational analysis

The size and geographical distribution of the organisation makes it difficult to check that each of the personal computer users are adhering to the organisation's distributed standards and guidelines.

As previously stated, the responsibility for personal computers rests with local departmental management and local Security Administrators. It is not practical to check on every user to ensure that standards are being followed. The main duty of the management is to ensure that each employee is aware of the need for information security and to provide the information as necessary to observe company standards.

Personal computer software was usually obtained from standard, reputable suppliers, but some arrived unsolicited through mailshots. Sometimes games were brought from home and loaded onto a company personal computer to play during lunch breaks.

The effects of a malicious virus spreading through the organisation's personal computers could have a significant effect on the computer based application systems. The biggest impact would probably be the time it would take to locate the spread of the virus, remove it and then recreate the data afterwards. The cost of such an operation is considered to be significant.

There have been no changes with regard to the roles and duties of personnel in so far as they relate to information security, however, positive steps (e.g. the installation of virus detection software on each PC) have been taken to reduce the likelihood of known personal computer viruses infecting the organisation's personal computers.

9) Reflections on the organisation

A memo from the company Managing Director was issued to every employee stating categorically that the playing and storing of computer games on company computers was now banned and anyone found to be doing this would be dealt with at a senior level.

The organisation has installed virus checkers on all personal computers. It has also established a 'quarantine area' for scanning any unsolicited software that may arrive through the post. Even though this service is well publicised, it relies on the active involvement of the recipient.

The organisation also accepts that there is a minor risk of being infected by a virus from a standard supplier but feels that the suppliers checks are adequate enough for this to be ignored.

10) What happened to the people concerned

An employee who was an Analyst/Programmer on secondment to the finance department was suspected. He was severely reprimanded, removed from the project he was working on and returned to head office.

The Analyst/Programmer left the organisation shortly after this incident to set himself up in business writing software for personal computers.

Members of management and those involved in IT security in the organisation were considered to have behaved correctly and in accordance with organisational procedures.

11) Conclusions from the case study

Viruses are considered by the organisation to be a real threat, not just to large companies but also to standalone applications in small enterprises and therefore should be taken seriously.

It was felt that the creation of standards and guidelines was vital and that it was essential to check for their compliance, even if it was only in a cursory way.

The organisation thought that awareness of the need for information security has to come from the top and each layer must be seen to support the policy.

Organisationally, information technology plays an important role in this company. In addition to business computing, it is used in other areas such as process control manufacturing and site security. Through experience, the company has a well-developed ethos of security towards its computing resources. However, its geographically dispersed nature has led to a fairly complex organisational structure with some duplication and some gaps in managerial responsibility.

In particular, the growth of PC based end-user computing has provided some weaknesses in overall organisational control. This is because the security ethos was based on centralised large scale computer systems. Consequently, there was a lack of sufficient security measures and procedures in this area.

The management of end-user computing resources is by its nature a difficult task. Its main purpose is to harness the creative talents of employees by arming them with powerful tools, but inevitably this can cause control problems from an overall organisational perspective. Managerially, it requires a level of trust combined with publicity and an educational policy that creates awareness of the organisational risks. This is largely because the disparate nature of the computing resources makes it very difficult to regulate each individual user.

Unfamiliarity with the culture of the PC can, as in this case, present problems. The ease of access and widespread use of PCs (many people own one at home) encourages the exchange and swapping of software, (especially games) and experience. This is completely in contrast to the bureaucratic, centralised and heavily controlled culture of centralised computing. The capabilities of many modern PCs easily outstrip mainframe computers of a few years ago and the sense of power accorded to end-users may lead to almost fanatical extremes.

The problems in the case arose out of a common blind spot exhibited by many computing professionals whose experience is based on centralised systems. Despite the existence of seemingly secure systems, they were unable to anticipate one of the major problems of PCs - virus infection.

The perpetrator of the misuse was clearly a PC enthusiast who was almost certainly using company resources for his own interests. His subsequent occupation as a games author would seem to confirm this.

An approach to secure systems such as that offered by the Virtual Methodology [Hitchings,1995] would not only reveal the organisational and managerial issues but it would expose this kind of weakness. It

000063

would also, for example, highlight the bureaucratic or political nature of control exercised by centralised computing departments and the clash with the ethos of end-user computing.

CONCLUDING REMARKS

The prevention of computer viruses and information security in general is usually considered a technical problem, however, it has become clear that organisational issues have been a major factor. The size and geographical distribution of an organisation can make information security weaker, especially if the company has become too devolved and there is no centralised checking or controls.

The ethos of organisations must be reconsidered with information security in mind. Too much trust is not desirable, it must be balanced with adequate controls. Allowing staff to run their own software on the company's machines may at first seem harmless. However, on closer inspection, it is obvious that a virus can enter a networked system in this way and spread throughout an organisation's computers.

Managerial issues are also of importance. Management skills should be improved so that a managers understand their staff and are aware of what they are doing.

It is now known that most threats to information security are from insiders. This highlights the fact that trust in employees must be balanced by adequate controls. It also shows that the myth of the lone hacker attempting to disrupt systems should not be the major concern of organisations.

By using a methodology like VM, information security can be dramatically improved because the entire organisation is considered and not just one area. Organisational ethos and policy are re-evaluated with security-in mind, as are managerial issues. It only takes one breach in security for a problem to occur and by looking at the organisation as a whole, as well as local issues, a successful information security policy is more likely.

REFERENCES

[1] Checkland, P., 'Systems thinking, systems practice', Chichester: Wiley, 1981

[2] Checkland, P. and Scholes, J., 'Soft systems methodology in action', Chichester: Wiley, 1990

[3] Wong, K. and Watt, S., 'Managing information security', Oxford: Elsevier Advanced Technology, 1990.

[4] Davis, D.W. and Price W.L., 'Security for computer networks', Chichester: John Wiley and Sons, 1987.

[5] Gasser, M., 'Building a secure computer system', New York: Van Nostrand Reinhold, 1988.

[6] Baskerville, R., 'Designing information system security', Chichester: Wiley, 1988.

[7] Hitchings, J. 'Achieving an integrated design: the way forward for information security', *Proceedings of the Eleventh International Security Conference IFIP SEC '95*, Cape Town, South Africa, 9 - 12 May 1995, Chapman & Hall.

FULLY AUTOMATED RESPONSE FOR IN THE WILD VIRUSES (FAR - ITW)

Mike Lambert

Frontier Corporation, 61 Coventry Avenue, Rochester, NY 14610, USA
Tel +1 716 777 4761 · Fax +1 716 423 9853 · Email mlambert@rochgte.fidonet.org

1 INTRODUCTION

I recently looked at some anti-virus (AV) products as a buyer and found that the state-of-the-art in virus response has moved forward, but only slightly, toward what we need in an enterprise environment. One product will auto-disinfect floppy disk boot sectors on presentation without user intervention. Another product will auto-recover from BS/MBR virus infections but still requires operator intervention. There is also a product that will install on infected systems and clean the system during the installation. There is a product that has been capturing virus specimens for years. Many products will notify someone over a network, but incident accounting seems to be missing altogether.

The reason for the slow progress may be because there is no vision of what kind of product we should be moving toward, or maybe the AV development community is just more resistant to change than other development communities. We can do something about the former; that is, tell the AV development community what we want. This paper is just that, the 'Enterprise Wish-list for AV product developers'. I hope that there will be other papers that will correct my errors or include my omissions when and if they are identified. Such work is good and will fill the need for the lack of visionary direction it seems that we need so desperately.

Generally speaking, we are still working with the philosophy of non-automated response to virus exposure and infection response. Basically it's 'product sees it, someone cleans it'. This is fine for the single user at home, but not much good for an enterprise environment. ***What we need is a Fully Automated Response (FAR) for virus exposures and infections produced by the viruses from which we are most at risk, the In The Wild (ITW) virus.*** I would like to see an automated response with no user intervention and automatic sample gathering and reporting for ITW virus infections and exposures. I think this is extremely important in our environments which are directly exposed to the same in-the-wild viruses on a daily basis.

FAR is a response philosophy for the known risk; it is not a substitute for unknown risk mitigation. FAR handles the 99% you know, not the 1% risk you have yet to experience. Once experienced, the unknown risk becomes the known risk and is included in the known risk handling (FAR).

2 SPECIFICATIONS AND DEFINITIONS

The target audience is the Corporate Security Manager or the Network Administrator responsible for Anti-Virus capabilities. This is not meant to be a feasibility study, a technical justification, or a technical description of implementation. This is intended to present the idea of FAR to the target audience, highlight a few problems which will surface, and describe what FAR might look like from the functional point of view.

This paper is concerned with only a subset of all known DOS viruses. That subset consists of the 'in-the-wild' (ITW) viruses. There is no provision for MAC viruses, or for viruses not found in the wild. The reader is charged to keep this in mind throughout the paper, because what is stated here and applies here applies to 'in-the-wild DOS viruses' only. What I am asserting may not be applicable in the theoretical realm which includes response for all known and future viruses, and it isn't meant to be.

In-the-wild viruses are those viruses which are actually cruising computers in homes and organizations. These are the viruses that are likely to visit your organization (99% or more of the time). There are zoos containing thousands of viruses. These zoos are passed around to professionals, non-professionals, and the curious. Just because a virus is in a zoo, it does not mean that you are likely to see it in your organization. 'Zoo specimens' should not be confused with 'in the wild' viruses. I will refer to the 'in the wild' as ITW.

The current document which attempts to identify ITW viruses and appears to be accepted by all is Joe Wells' *Wildlist*. It is a good starting place, but one should keep in mind that there are viruses in the wild which are not on Joe's Wildlist, and a few problems with the list itself. For instance, because of naming variations, the same virus appears more than once. Since there is not a repository for actual ITW samples, some of the listed viruses are too vague to point to a specific strain. I think using this list is the most acceptable place to start, and that the list will overcome its current problems as it is used more.

I use the term 'virus exposure' as the general definition of a clean system in which a virus is present. Examples of this are: 1) a clean system which has an infected floppy disk in the drive; and, 2) a clean system which has an object (file infector) with a virus in it that, if executed, can infect the system.

I use the term 'virus infection' as the general definition of a system which has a memory-resident virus active and capable of reproducing (non-resident viruses not included).

I use the terms 'enterprise', 'organization', and 'site' to indicate a networked system with multiple DOS computers connected to it. This could be a small company, a school or university, or a multi-national corporation.

FAR is Fully Automated Response. This is an automated identification, disinfection, and reporting response to those viruses for which it is most appropriate. No user intervention is required and no user notification is delivered. Most BS/MBR, multipartite, and file viruses can be dealt with by restoring infected objects, even with the virus present. Some viruses cannot or should not be included in an automated response; these viruses are termed *non-FAR viruses*. Non-FAR viruses include those which destroy the executable or which require the virus to be present to access data.

A *FAR-ITW virus* is an 'in the wild' virus that can be disinfected without a special disinfection procedure (i.e.. can be totally contained in a software solution). All ITW boot sector (BS), master boot record (MBR), companion, directory, appending, and prepending viruses that do not produce 'virus resident dependent' problems should be included in a fully automated restoration capability. Exceptional ITW viruses requiring special disinfection procedures need not be included in the FAR requirement and are noted as non-FAR-ITW viruses. EXEbug would be a FAR-ITW virus, One-half may be a non FAR-ITW virus (depends on the expertise of the AV developer). Some hardware implementations may dictate some virus infections as non-FAR.

A 'workplace interruption' is any unnecessary interruption of a user's productivity. This includes manually cleaning floppy disks, scanning systems, cleaning systems, etc. A workplace interruption is not limited to a single worker; it often extends to other workers nearby. Workplace interruptions can be short or extended. The organization loses productivity with each incident which interrupts the worker. This productivity loss is often much more expensive than the actual cost of the technical response to the incident. In larger organizations, with employees located across large geographical areas, virus incident support often requires the user, increasing the magnitude of the interruption. An interruption need only occur when the ITW virus infection is non-FAR.

Non-ITW viruses can be included in the FAR but if it is not ITW there is little reason to include them. Doing this work ahead of time will certainly be necessary if the virus ever attains the ITW status. If a non-ITW virus is subsequently found in the wild, it can be included at that time.

No specific network is required for this model; communication requirements are stated generically. The specific implementation will vary from network to network, feature to feature.

'NLMs' and FAR have almost nothing to do with one another. File and multi-partite infections may reside on servers, but it is workstations that 'get infected' and 'spread infections'. An NLM does not FAR; don't confuse an NLM scanning executables delivered from a server with a Fully Automated Response to a virus exposure or infection. The closest an NLM comes to FAR is to 'move' or copy a suspected file to a protected directory. This is not FAR. You may need an NLM if you don't have any decent network security (i.e. someone using the network can infect an object residing on the network), but FAR is a different philosophy.

3 IN THE WILD VIRUSES

So just how many of the thousands of viruses are we talking about? It is not near as many as you think! Joe Wells' list of July 1, 1994 lists a total of 152. January 1, 1995 shows a total of 197. The current list of June 1, 1995 gives us a total of 235 worldwide! I know that there are more ITW viruses than those on Joe's list. It is the requirement of list participation which causes this problem. Even if we add 30% to Joe's list we still only reach just over 300 viruses!

Of the 6,000 or more viruses in existence, it is most probably a mere 300 that we are talking about for FAR! This is just 5% of the world's virus population.

Let's say that 35% of the ITW are BS/MBR viruses. A conservative figure is that 65% of the virus incidents in the an organization are ITW BS/MBR viruses. This means that more than 65% of the problem is caused by a mere one and one quarter percent (1.25%) of the viruses! In many organizations, BS/MBR infections account for 90+% of the number of virus incidents.

I think that when we look at the viruses which we see every day; they are virtually all ITW viruses.

4 THE FULLY AUTOMATED RESPONSE (FAR) OVERVIEW

Fully Automated Response is:

A. Detect the ITW virus

B. Remove the ITW virus automatically and without any user notification or intervention

C. Report the incident to technical support

The difference between current solutions and FAR is that software does all the work automatically and quietly. We make all the decisions up front.

Decision 1 - always remove the ITW virus.

Decision 2 - always report incidents and include a sample of the virus.

Decision 3 - always compile statistics.

The event, as an exposure or an infection, must be closed by the conclusion of the response. After-action decisions, such as notification, are handled outside of the response. *It is this 'total response to the event' that we currently lack.* We augment our current responses with customer service manpower, technical support manpower, user manpower, and training manpower. All of this manpower is a waste of our organization's resources. We've just got to do things smarter!

<u>FAR should be considered a 'medium security' solution.</u> (Those needing a 'high security' solution must add appropriate technology.)

Note that FAR lacks the 'tell the user and make a big deal of it' philosophy. I have found in asking users that the last thing they want to deal with is a virus; 'I just want to do my work' is the most common phrase I hear when talking to enterprise users. I agree with the user; let them work. Let virus administrators notify users when necessary.

For those concerned with 'user interfaces', what better user interface could there be than FAR? *Absolutely no user input is required.* In non-FAR situations, user response is via a message directing the user to call the Help Desk (or whatever is desired). Administrator interface (to the FAR product administration) is a different issue, and keep in mind that system administrators generally are more computer literate than most end users.

The reasons we need FAR are:

- *It costs too much* to 'open a ticket and dispatch a tech' for individual virus incidents
- The resulting (and expensive) *workplace interruption is not necessary and too costly*
- We can *cut down the necessary virus training* for technicians and employees
- We *need the automated reporting,* so that we can justify re-licensing the product next year.

If your virus problem consists solely of infections by FAR-ITW viruses, it is easy to calculate the cost savings. Just add all the costs of the Help Desk calls, plus the PC technician's time, plus the lost worker productivity. This is the amount you would save with FAR for ITW viruses.

FAR need only handle our biggest risk, the ITW virus. There is no need to include every obscure research virus in the FAR concept. New ITW will viruses appear, and need to be included in the FAR. Most products have demonstrated that they can supply scanners and cleaners for new ITW viruses very quickly (sometimes in mere hours), so including them in FAR products is trivial.

The idea behind FAR is to use an automated response to a FAR-ITW virus infection or exposure <u>whenever possible</u>. *This includes product installation.* There are no technical reasons for why a FAR-ITW virus cannot be dealt with when installing a FAR product on an infected system. If generic solutions such as a generic MBR restoration instead of a real MBR restoration are necessary, these are acceptable.

'False Positives' are non-existent in the FAR model. Since we are talking about working with known viruses, both memory and object infections can be positively identified.

FAR is not a replacement for all current AV products. FAR is the distillation of the best technologies into the exact product that will be of the most value in fighting viruses on the front lines. FAR's technology comes from the multitude of other products, <u>all FAR necessary technology is already in existence</u> in different products. Research and Development of current products must continue. There is no reason for

FAR to be the only technology an organization utilizes. All organizations must select the technology which best fulfills their security philosophy.

5 PROBLEMS WE FACE TO FAR-ITW

We face four basic problems to implement FAR-ITW:

- No ITW base from which to start from
- Current 'zoo' certifications
- Industry resistance
- Ourselves.

NO ITW BASE

Unfortunately, while Joe's Wildlist is a starting place, it is not a base to define the ITW. There is no actual 'ITW zoo' from which to specify 'these are the ITW viruses'. The only public ITW attempt is currently made by *Virus Bulletin* and it does not resemble Joe's list to any great degree. I have 90% of the ITW viruses on Joe's list and it is easy to argue that they are not the actual ITW viruses which are reported on the list. This is due to the variety of names and inexact identification by some products. Still, it is from this list that we can at least start an interim solution.

We need two solutions to this problem.

The *first* is an interim solution which will comprise a consensus of what AV developers and professionals will concede that makes a reasonable ITW zoo. This work is in progress by Richard Ford at the *NCSA*. Richard is working with the Anti Virus Product Developer group to define an initial ITW specification. The agreed-upon samples will form the first ITW sample base.

The *second*, long-term solution is to 'start from scratch'. We must assemble a 'new' ITW list directly supported by an ITW sample base. These samples would be from actual ITW infections. This would provide the indisputable base from which to launch a definite FAR-ITW implementation. Information compiled must include generic site information and specific virus information *for each infection and exposure reported*. This will give us more information than ever, virus location and prevalence. All incidents need to be reported to get an accurate picture of the true ITW virus.

We, as corporations, organizations, sites and private users, must assist in the assembly of this new base by reporting ITW infections and exposures, complete with samples. The new ITW sample base will identify a real sample of the actual ITW threat to computer systems that all can agree on. This must be a world-wide effort. I am confident that the AV developer, professional, independent, and the user communities will support such an effort.

To facilitate such a massive effort requires central clearing houses from which to assemble reports and samples to define the actual ITW virus. Existing and new reporting lines will be able to support and represent all of us. Joe Wells has a reporting structure in place. Joe will collect samples. *Virus Bulletin* has been reporting ITW infections and can collect samples. The *NCSA* has a large number of members and being a security organization seems the most likely place to house the ITW sample base. All ITW samples must be made available to all AV developers (a situation that does not now exist). Other professionals and independents in organizations can help by forwarding reports and samples to another central location. This could be the first joint venture for all to unite to solve our biggest single problem.

The actual details of creating a working solution from which we may all come together is in the making. It looks like there will be three or four different places to report infections and exposures and send samples. More details will be forthcoming.

ZOO CERTIFICATIONS

While there have been no ITW sample bases to test products with, there are a lot of 'virus zoos' out there. Using these zoos to 'test scanners' has been the definition of AV product testing. 'Zoo scanning' requires a known control base and proper interpretation of results, two things in short supply. What has taken the place of representing our actual threat has been this 'zoo testing'. The scanner is pitted against one or many private collections of a myriad of viruses, Trojans, joke programs, simulator samples, and just plain junk files. We are 'given results' and told that this certifies some product as good for use in our ITW environments. Few things are further than the truth.

Zoo certifications are really a sort of 'what if the research virus made it to the wild?'. Occasionally this happens, but frequently the new ITW virus is new, so 'zoo scanning' doesn't help. Proper interpretation of this sort of 'scanner testing' may be valuable, but just isn't directly transferable to solutions to our ITW problems. Worse still is that these certifications give one a false sense of security because they are very inadequate tests. Just because a scanner can identify something in a zoo scanning test does not mean that it will find the virus in an infected environment (this is a common occurrence).

We *must* de-emphasize 'zoo certifications' and emphasize 'ITW certifications' using a known ITW sample base. Casual magazine-type of 'AV testing' without professional guidance must be avoided.

INDUSTRY RESISTANCE

The AV development industry has just come out of a war with the virus creation community. The virus creation community provided, the AV development community included, regardless. This has little to do with our ITW problems except that some AV developers got so caught up in the war, they forgot the civilians. The fact is, until recently when ITW virus testing became something which could not be ignored, it was ignored by most. There has never been a concerted effort by the AV community to deal with ITW viruses. AV developers must engage 'full disclosure' of ITW viruses. Current exchange restrictions should only apply to research viruses. Current systems and philosophies will take time to change.

The AV industry is sometimes inflexible in what we need, but inclined to what they think we need. Consumers bear a fair amount of the blame for picking products which identified ever-increasing numbers of viruses (zoo certifications) and insisting on a scanner solution. We have refused to listen to sage advice for the different level of security solutions, insisting that there should be just one, the scanner.

The industry is geared to our marketing weaknesses and will resist the direction in which we really need to go. We must change our requirements to reflect our real needs if we wish to move the AV industry to provide the solutions we need.

OURSELVES

We as a consumer community are going to have to look at the information that we see and make some intelligent, objective analysis. We have to learn the difference between an article and an advertisement. We are considered by the marketing community to be drones waiting to be told what to do. Don't blame anyone else; it is our fault. We are much more educated in other areas of purchase, and naïve in AV product selection.

We need to get reliable information from security organizations rather than advertisements. We need to support activities which are directed at ITW viruses and the reliable, appropriate handling of them. If your security organization does not have sound, measurable efforts in these areas, we need to demand them.

6 THE ANATOMY OF FAR

FAR is comprised of six tasks:

1. Identify the ITW virus
2. Collect a sample
3. Clean the ITW virus
4. Make a report
5. Send report to virus administrator
6. Keep all incident statistics for the virus administrator

and must operate equally well in:

1. The clean environment
2. The infected environment

Under clean conditions, FAR requires that the virus be cleaned without user notification or intervention. Notification of the exposure should be a virus sample and report to the virus administrator. Leave the user notification to the virus administrator. Installation should be automatic, no user intervention required, and should save the MBR, BS, system files, and command processor (depending on product design philosophy). These objects can be used later, in infection recovery instances.

Under FAR-ITW infected conditions, FAR requires that the virus be handled exactly as under clean conditions (i.e. identify, remove, report). Installation should recover the system and save the same objects as under clean installation.

Under non-FAR-ITW infected conditions, FAR requires that the user be notified to contact his local technical support personnel. Under these conditions, the situation is hazardous enough to warrant the workplace interruption. All of the sample gathering and reporting should be done, but in the non-FAR-ITW infected condition, the virus is not automatically cleaned.

All of the techniques necessary to accomplish the related tasks necessary for FAR have been in use in one product or another at some time. I note that some products really excel in some of these areas. What we have never seen is those techniques combined to accomplish FAR. It's not that FAR is impossible; it just hasn't been a priority worth pursuing...yet. If the 'best identification', the best 'working in an infected environment', and the 'best restoration' were all to join together with the 'best accounting', we could have a superior FAR product!

7 FAR IN OPERATION

To be complete, I must describe FAR response enough to facilitate its development. Some managers may not be concerned with some of the specifics. Skip this section if you wish.

A THE INFECTED FLOPPY VIRUS EXPOSURE

This is the easiest response to implement. The object in question is not executed and needs to be replaced with a known object.

FAR says that, when a user inserts an infected floppy disk in the system, the following happens:

- Identify the ITW virus

- Save a sample of the BS
- Clean the floppy if not write-protected
- Create an infection report
- Send the report and sample to the virus administrator when possible
- Integrate the infection report into the virus administrator's summary.

Several things happen and don't happen in this instance:

- The user continues working with a clean floppy which will not later infect a computer
- The administrator knows what the user is exposed to before the organization gets infected by it
- Virus incident statistics are automatically compiled

I do not consider the failure to restore a BS on a write protected disk as a reason to interrupt the user. This disinfection failure should be forwarded to the virus administrator and the appropriate decisions made by the administrator.

What's left to do? The virus administrator needs to determine how and when the user is notified of the problem which could have disrupted their work. The administrator also needs to determine the fate of the collected sample.

B THE BS/MBR VIRUS-INFECTED SYSTEM

By definition, this should not happen on a FAR-protected system. Suppose, for this example, that someone booted an infected floppy while the regular user was on vacation.

The FAR response is:

1. Identify the ITW virus
2. Save a sample of the MBR or BS
3. Clean the MBR or BS
4. Reboot the clean system
5. Create an infection report
6. Send the report and sample to the virus administrator when possible
7. Integrate the infection report into the virus administrator's summary

Again, the user continues to work without an interruption, the administrator knows what is happening, and statistics are gathered.

C THE FILE VIRUS EXPOSURE

This looks much like the response to the exposure to the infected floppy BS, but additionally must deal with an object that must be executed. There will be cases where the 'clean the virus' requirement may not be able to be fulfilled. If this situation exists, all other FAR requirements should be fulfilled.

- Identify the ITW virus on copy, execution, etc
- Save a sample of the file
- Clean the object if not write-protected
- Perform the originally requested action (copy, execute, etc) if possible

- Create an infection report
- Send the report and sample to the virus administrator when possible
- Integrate the infection report into the virus administrator's summary

Several things happen and don't happen in this instance:

- The user continues working with a clean executable that does not later infect the system
- The administrator knows what the user is exposed to before the organization gets infected by it
- Virus incident statistics are automatically compiled.

I do not consider the failure to restore a write-protected file as a reason to interrupt the user. This disinfection failure should be forwarded to the virus administrator, and the appropriate decisions made by the administrator.

Same cleanup as the BS exposure: The virus administrator needs to determine how and when the user is notified of the problem that could have disrupted their work. The administrator also needs to determine the fate of the collected sample.

D THE FILE VIRUS INFECTION

File infections come in a huge variety of infection types and techniques. In some cases, the virus is non-resident, making it easy to deal with. In others it may be prudent to use the resident virus. Some viruses are so virulent that they must be removed from memory entirely. The exact technique or series of steps to disinfect the system is dictated by the virus type and capability. However, the goal is the same: remove the virus without user intervention, get a sample, and make a report.

In this case, we imagine an executable which is infected without FAR protection. The subsequent starting of this system thus makes the virus resident. Let's say someone booted their own floppy, infected the system, and left. The virus infection will be active when the system is subsequently booted by the regular user.

The general FAR response is:

- Identify the ITW virus infection
- Establish a clean environment or change configuration, if desired or necessary
- Secure a sample of the virus
- Restore the infected objects or delete companions
- Create the infection report
- Restore the system to an operational configuration if necessary
- Start or restart the system
- Send the report and sample to the virus administrator when possible
- Integrate the infection report into the virus administrator's summary.

The order of some items can be changed depending on the implementation philosophy. The point is to get a sample, remove the virus, and notify the appropriate party when possible.

While the user may get a little show as the system is restoring itself, the user goes to work with the minimum interruption once the restoration is completed, the administrator knows what is happening, and statistics are gathered.

E THE MULTIPARTITE VIRUS EXPOSURE AND INFECTION

The FAR response for the multipartite is much like the BS and file exposure and infection responses, with the added requirement of the additional object restoration.

Again, the user continues to work without an interruption, the administrator knows what is happening, and statistics are gathered.

8 WHEN NOT TO FAR

Not doing FAR does not mean not doing *all* of FAR tasks. Whatever steps can be done, should be done. In many cases, this requires notifying the user and possibly creating an interruption. If nothing else, there should be a sample gathered and report created. The report can be sent if or when practical. If there must be a workplace interruption, the user can be directed to call the Help Desk with a message.

There are times when you don't want to FAR. This would be when removing the virus would deny access to data (in whatever form this may take). In this case it may be necessary to back up data while the virus is present and then remove the virus. Years ago, this concept was illustrated by the Volga virus. It's not that it is impossible to FAR with Volga or One half, there just may not be enough use of the particular FAR technique to warrant the expense of developing it.

Another instance is when your FAR product meets an anti-AV product that is FAR hostile. Theoretically this should not happen, as the FAR response should be to remove the virus without 'setting off the bomb'.

Other non-FAR situations are those where the object is write-protected. This can be a file on the server, a write-protected floppy, or write protection on the workstation. These situations are non-FAR as cleaning the object is concerned, but should still be as FAR handled as possible.

There are other non-FAR situations that can be described. AV developers may decide to make different viruses FAR and non-FAR. *The definition of a non-FAR virus is 'that virus for which no AV developer can provide a software only solution'.* Not all AV developers are created equally. I wouldn't be surprised to see some 'partial-FAR solutions' for those which lack the skills for a complete solution. I'm sure we will see some viruses considered non-FAR because the virus is not sufficiently in the wild to warrant the work necessary for the FAR solution. There are many shades of gray which the tailor may use.

9 FAR BENEFITS

Of course the benefits are obvious. The enterprise has software do all the work, and the security and network administrators get all the credit. The software even justifies itself for you. The benefit for AV developers is new product potential, significantly less user technical support required, and enterprise users have the justification to relicense next year.

10 FAR EVALUATION

I hope that FAR product evaluation does not parallel many of the current testing and evaluations. If the evaluation does not make a good case for the utility of the product, and properly evaluate that product, we should disregard the test. FAR testing should primarily be concerned with the ability of the product to do the job (and the job is viruses). Tests which are not primarily concerned with the ability of the product to do the job for which it was designed should be ignored.

We, the community which needs and will use FAR, should require that all primary testing is geared to the security aspect of the product, dealing with ITW viruses. Other tests for ease of use for administrators should be clearly labeled as secondary tests, not directly measuring the product for its primary job of dealing with ITW viruses. All Objective conclusions pertain to the primary job of the product; Subjective conclusions pertain to the secondary job of the product.

11 CONCLUSION

FAR is what users want.

FAR saves organizational resources (money) responding to virus incidents.

FAR makes almost all viruses in the enterprise environment a non-event.

FAR takes no technology other than what is already in use to implement.

FAR will be a reality when we demand it.

The question is not 'do we need it?'; the question is 'when will it be provided?'. I'll bet the first AV developer which produces a working version will find that it is 'the better mousetrap'. Just open the door.

CREDITS

PC Viruses in the Wild	(Wildlist) is a collation by Joe Wells in co-operation with many AV product developers and AV professionals.

The ideas and opinions expressed are wholly my own and are not necessarily those of my employer or associates. I wish to acknowledge and thank all those people who have contributed to the AV community and its advancements; their efforts and views become a part of all of us and are many times difficult to separate from our opinions.

Special thanks to my friends and associates who have listened, assisted, and criticized me during the writing of this paper. The remaining imperfections here are mine alone.

THE PC BOOT SEQUENCE, ITS RISKS AND OPPORTUNITIES

(THE USE AND IMPORTANCE OF THE BOOT SEQUENCE IN REDUCING BOOT VIRUS EPIDEMICS)

Jonathan D. Lettvin

OverByte Corporation, 194 Waltham Street, Lexington, MA 02173-4914, USA
Tel +1 617 860 9119 · Email jonathan_lettvin.lotus@crd.lotus.com

ABSTRACT

A single bad user habit accounts for the epidemic spread of most boot sector viruses (leaving diskettes in the boot drive). Users need education to 'unlearn' this habit. The best time to educate users is when they boot.

No general anti-virus product provides boot-time educational text, or anti-virus code. Some products provide a virus detecting TSR for DOS. This would usually be ideal because the TSR identifies a boot sector virus before it is activated by finding it on the floppy before the user boots. However, lack of compliance, bad habits, and unavoidable events make booting from floppy a serious continued source of virus spread.

We have discovered that specially prepared diskettes, formatted for general use and for delivery of commercial products, provide a new and important layer of virus epidemic prevention.

WHO AM I?

I am an employee of *Lotus Development Corporation* and, at the same time, I am president of *OverByte Corporation* having a special relationship with *Lotus*. *Lotus* uses our anti-virus products and services. I speak to you today as the President of *OverByte*.

WHO CARES ABOUT EPIDEMIC BOOT SECTOR VIRUSES?

I think everyone in this room has spent time dealing with epidemic boot sector viruses. The 'virus prevalence charts' document that epidemic boot sector viruses dominate the virus industry. If you are an anti-virus developer or reporter, you are confronted daily by epidemic boot sector viruses. The majority of corporate anti-virus dollars are spent on removing epidemic boot sector viruses.

I've spent over six years at *Lotus* studying viruses and the corporate response to the problem. In the last five years, I saw exactly one 'file infector' attack by Natas. All other virus attacks were attacks by the top twenty or so epidemic boot sector viruses. At *OverByte* we know customers need to detect thousands of other viruses just for diligence, but we rarely hear of even those thousands of viruses that are in the wild.

WHO CARES ABOUT THE BOOT SEQUENCE?

I believe the moment of virus attack is the best time for virus defense. At no other time may any program assume total control of a PC without risk. The boot sequence is unique in its opportunities for virus combat. All epidemic boot sector viruses rely on special knowledge of the boot sequence. Boot sector viruses modify BIOS resources before any operating systems are loaded. Once an operating system is loaded, restoring BIOS resources perfectly is impossible.

When a virus is detected, many anti-virus products force a *clean* reboot of the PC as part of virus removal. The methods used by *OverByte* to restore the boot sequence often pre-empts the need for rebooting.

I feel that this is important. I believe that you in the audience will agree with me when you experience the difference DisQuick diskettes make both in initial virus detection time within a group, and the elapsed time between an attack of an epidemic boot sector virus and when you can get back to work.

WHAT IS THE BOOT SEQUENCE?

This talk is focused on what I have identified as the boot sequence. I consider the boot sequence as the transition from Power-Up, through POST (Power On Self Test), MBR, and BS to any OS (Operating System). Virus programmers have exploited this transition more than any other aspect of the PC. I will certainly not be able to cover the subject exhaustively during this talk. Mostly, I will describe a few of the methods already exploited by virus programmers and mention quickly other areas needing protection.

I know of no good book describing the exact requirements of the boot sequence. I constructed the 'GENERIC PC STANDARD', which we will describe shortly, and how it is used by the boot sequence from first-hand experience and hints from the many popular books which cover PC Hardware, BIOS, DOS, and collateral subjects. I have put a short bibliography of our most frequently referenced books at the end of this paper.

WHAT IS THE GENERIC PC STANDARD?

What defines a generic PC is a fairly uniform standard of operation between hardware and firmware. This standard is so well documented that very little stands unrevealed, although almost none of it is official. The standard is so firmly entrenched that a specific PC has little market unless it scores very close to 100% against compatibility tests.

Compatibility tests include recognition of hard-numbered BIOS entry points for vectors, exact contents of certain RAM locations during the POST (Power On Self Test) and boot process, exact contents of certain disk locations, and many, many more. The actual standard even allows for relaxation of enforcement on documented standards. This means that methods considered to be permanent standards often change.

One example of an item incorporated in the GENERIC PC STANDARD is that the absolute real-mode ROM-BIOS location F000:EC59 will probably remain permanently as the entry point for floppy diskette BIOS services. Many useful programs use this hard-coded entry-point.

To clarify a little more, consider what a generic PC is not. It is not the applications we run every day. It is not the device drivers loaded by our preferred operating systems. It is not DOS, OS/2, or Windows. However, almost all of these rely on the GENERIC PC STANDARD, and will fail on PCs which deviate from this norm. It is not even the specific PC on your desk which has a specific configuration and physical options set, although, in all likelihood your PC follows the generic PC standard completely.

WHAT ARE THE RISKS IN DEVIATION FROM THE STANDARD?

Deviation from the standard can cause failure of operating systems, device drivers, and application programs. Developers of add-in cards must usually recognize and follow these standards completely to be successful.

On a system where a standard breaking card is installed, viruses sometimes fail, with dramatic results. Even on a system where the standard has been upgraded, viruses which depended on an earlier standard may cause system failure. For instance, some common viruses do not recognize the 1.44 MB drives which are now the standard boot drive. These viruses make the assumption that the drive is 720K, and use unintended sectors for propagation.

Most viruses are not successful in following the new standard for operating systems which engage 'Protect Mode' on the newer *Intel* chip family members. These operating systems will sometimes warn the user with obscure references like 'Cannot load 32 bit driver', when a virus has chained a vector which needs to be in ROM-BIOS to follow the new standard.

I suspect that part of the reluctance in adopting new operating systems has been due to interference by viruses, making a given PC appear to ignore the new standard, and thus misbehave. This misbehaviour is often seen as a dramatic decrease from the intended performance of the new operating system. This kind of anecdotal report spreads rapidly by word of mouth, making potential new users reluctant to purchase otherwise fine software.

WHY ARE ANY VIRUSES SUCCESSFUL AT ALL?

Not surprisingly, most successful epidemic viruses attack PCs as hardware. Most virus programmers use knowledge of the GENERIC PC STANDARD, as well as knowledge of specific file systems and operating systems, to propagate. Since the average user is interested only in application programs useful to their daily activities, the very slight changes viruses make in resources and operations on which the operating system relies are usually either invisible or merely nettlesome until the warhead triggers.

WHY IS THE FORM VIRUS THE MOST SUCCESSFUL?

Probably the most frequently found virus in the wild is the 'Form' virus. We believe the reason for its success is its very careful attendance to the PC and FAT file system standards. I note the remarkable care taken by its author to allow Form to adapt. Form appears to lack a formal warhead, but it has certain technical flaws which amount to an accidental warhead. I could be convinced that the flaws were intentional. The consistency of code style breaks for this flaw.

WHAT CHARACTERIZES BOOT-TIME RAM AND VECTORS?

The actual condition of the vectors and BIOS data when the boot sector has been read in is required to have a certain character. When this character is not correct, we believe a number of methods can be used to correct it. The only time this character may be corrected safely is during the boot.

For example, a documented BIOS vector will point into one of several known areas. If the vector is in a segment F000 or above, we consider it safe. If the vector is in segment C000 or above, but below D000, we consider it safe. If the vector is below segment A000, we consider it distinctly unsafe. However, the standard leaves segments D000 through EFFF as areas of conjecture. We have methods for dealing with these as well, but the standard is unclear for that range.

WHAT ARE THE STANDARD STEPS IN THE BOOT SEQUENCE?

The PC boot sequence is a long process to explain in words. I will be discussing the four numbered items of the boot sequence in greater detail after the description of the entire boot sequence in digested form. I break the sequence down into four major sections:

NORMAL PREAMBLE TO FLOPPY OR FIXED DISK BOOT

Starting at power-on, the Intel chip loads REAL CS:IP with FFFF:0000.
This location is in ROM-BIOS (Writable FLASH EPROMS in newer PCs).
The instructions at FFFF:0000 start the Power On Self Test (POST).

1. POST installs vectors pointing into ROM or shadow RAM.
   POST performs all its duties and scans for additional ROMS.
   Each ROM may initialize more vectors, returning control to POST.
2. CMOS memory is examined to determine the device to be used for BOOT.POST finishes and checks for presence of a floppy in the BOOT drive.
   If a CMOS confirmed floppy is present, BIOS performs the FLOPPY BOOT.
   If not, BIOS performs the FIXED DISK MBR BOOT.

NORMAL FLOPPY BOOT

Read drive 0, sector 1, head 0, track 0 into location 0000:7C00.
Then continue the process with the GENERIC OPERATING SYSTEM BOOT .

NORMAL FIXED DISK BOOT

Read drive 80, sector 1, head 0, track 0 into location 0000:7C00.
Set CS:IP to 0000:7C00.

3. The code executed is usually a standard MBR.
   The MBR contains two critical data areas, one undocumented.
   DRIVE ID used by newer Microsoft Operating Systems
       Offset 1B8H through 1BEH is undocumented.
   PARTITION TABLE (PT)
       Offset 1BEH through 1FEH is well documented.
   The MBR code starts by copying itself to location 0000:0060H.
   The code sets CS:IP to 0000:0060H transferring control to the copy.
   The MBR code reports any errors or absence of partitions in the PT.
   If there are no errors, the MBR code loads the active partition BS.
   This BS is loaded at 0000:7C00.
   Continue the process with the GENERIC OPERATING SYSTEM BOOT.

GENERIC OPERATING SYSTEM BOOT (USING IBM DOS AS AN EXAMPLE)

Set CS:IP to 0000:7C00.

4. The code executed is usually a DOS BOOTOR SECTOR sector.

   This code may change diskette-tuning parameters in BIOS.

   This code analyses the BPB (BIOS Parameter Block) for disk structure.

   The BPB contains the size of the system areas and the sector count.

   Using the BPB, the directory is found and loaded.

   Compare the first items in the directory to IBMBIO.SYS and IBMDOS.SYS.

   If they exist, set the cylinder/head/sector to logical cluster 2.

   Calculate the cluster count from the size of IBMBIO.SYS.

   Load those sequential clusters starting at cluster 2 into RAM.

   If absent, notify the user of a 'Non-System Disk' and request reboot sector sector sector sector. If present, set CS:IP to the beginning of the loaded IBMBIO.SYS.

For brevity, I have marked only four of the many places of virus risk in the sequence. These four will be discussed next.

WHAT ARE SOME PRIMARY RESOURCES EXPLOITED BY EPIDEMIC BOOT VIRUSES?

1. VECTOR CHAINING

Most boot sector viruses simply chain INT 13H (BIOS disk services). Some boot sector viruses chain a vector like the timer and wait for DOS to be loaded. All current epidemic boot sector viruses occupy RAM and not ROM. This may change, due to certain advances in PC BIOS distribution.

2. CMOS MODIFICATION

Some viruses will modify CMOS RAM during infection, and force the PC to ignore the diskette drives during subsequent boot. This kind of virus will then always boot from fixed disk. Once the virus boots, it may hide its having booted from fixed disk by detecting, loading, and executing the floppy boot.

3. MASTER BOOT SECTOR CHAINING

Many methods are used by viruses to gain control during fixed disk boot. One virus has completely rewritten the code for loading the active partition. Others use methods similar to diskette boot sector chaining. Some viruses change methods on fixed disk, when the same method would suffice.

4. OPERATING SYSTEM BOOT SECTOR CHAINING

We see two common methods for storing boot sector viruses on a floppy. The first method moves the original boot sector to a fixed location. The second method calculates a place to store the original boot sector. When using the fixed location, the virus programmer counts on rare overwrites. When using calculation, the virus programmer counts on rare disk maintenance. Both methods have been successful. An attempt to subsume the entire legitimate boot code will probably fail.

WHAT ARE SOME BOOT-TIME OPPORTUNITIES AGAINST EPIDEMIC BOOT SECTOR VIRUSES?

1. VECTOR CHAINING

Certain portions of the first 1 MB are known to be ROM only. Other portions may be either RAM or ROM, and still others are RAM only. We detect vectors which are potentially pointing into virus-owned memory.

2. CMOS MODIFICATION

Where a specific virus has modified the CMOS memory, the original values can usually be restored.

3. MASTER BOOT SECTOR CHAINING

We cannot claim to protect Master Boot Sector code from rewriting viruses. However, in combination with our diskette methods, we will still detect them. For chaining viruses, our code will usually alert the user to a virus attack. We will install a new and better Master Boot Sector code portion than the original when we remove one of these viruses.

4. OPERATING SYSTEM BOOT SECTOR CHAINING

BIOS has a documented unchangeable diskette vector which can be safely used. Certain precautions must be taken to guarantee that safety. With the virus disabled, the vector chain through the virus may also be used.

WHAT ARE THE 24 CRITERIA BY WHICH WE MEASURE ANTI-VIRUS PRODUCTS? (MORE OR LESS IN THE ORDER OF EXPECTED OCCURRENCE)

We feel that epidemic boot sector viruses should be given special attention, beyond simple removal. Over time, we have developed a set of criteria by which we measure our own and other anti-virus products. We currently have 24 such criteria.

Some criteria measure solutions to technical problems. For example, all viruses can be disabled while they are active in RAM. We expect anti-virus products to disable at least the epidemic viruses.

Other criteria measure solutions to social and operational problems. For instance, we believe the most effective time to educate a virus victim is the exact time at which they put themselves at risk. The action which put them at risk must be defined simply, and the ways of avoiding future risk must be explained, also simply.

Eight items in the following list will be given extra attention during the talk. They are marked by asterisks (*).

* 1 **Detection**

There is a virus here.

* 2 **Identification**

It is this virus, with sufficient certainty.

3 **Acquire pre-removal recovery data**

Recovery is sometimes difficult without the virus present.

* 4 **Disable viruses**

Chain through all virus ISRs, or replace vectors.

000083

5 **Report virus disabling to user and identify the virus**
Let the user know when the virus function has been stopped.

6 **Overwrite the virus in memory**
Leave ISR chains when required to guarantee operations.

7 **Restore/replace boot sector/master boot record as necessary**
Restore when a copy has been saved, replace when overwritten.

* 8 **Offer a new Master Boot Record even when no virus is present**
Only install new MBR to detect future viruses if user accepts.

* 9 **Overwrite virus data on disk with removal distinguishing data**
Users must be able to identify overwritten data easily.

10 **Restore CMOS RAM data where possible**
Some viruses protect themselves by preventing floppy boot.

* 11 **Report damaged files if possible**
Decode the FAT and report files overwritten by boot sector viruses.

12 **Restore program files where possible, back up ambiguous restorations**
Leave the choice to use the infected file up to the user.

13 **Report infected files**
Believe it or not, some products do not report properly.

14 **Report virus removals**
This too is sometimes inadequately reported.

* 15 **Describe how this virus infects PCs, and how to avoid re-infection**
The user must be educated to avoid re-infection.

16 **Provide virus Hot-Line information customizable for MIS departments**
Most companies want an internal specialist to be notified.

17 **Announce product name, version, and copyright**
Everyone does this correctly nowadays

18 **Perform all these functions optimized for speed**
Some anti-virus products are slow and/or cumbersome.

19 **Store encoded session results on original media if write-enabled**
Keep a very condensed record of results: there may be many.

20 **Allow user to write-enable for storing encoded session results**
Give users a chance to recover from a simple mistake.

21 **Offer to copy contents from virus-infected floppy to clean one**
Users need to continue using their current active diskettes.

22 **Suggest write-protection for current diskette (compliment if found)**
Keep encoded session safe for later review.

23 **Provide session results to caller**
For desk-to-desk virus removal, an easy review method is good.

* 24 **Restore RAM (if boot loaded) or suggest reboot (if COMMAND loaded)**
Bring the PC back to the GENERIC STANDARD for the OS.

We have developed these 24 criteria as a guideline. Each point helps us produce high-quality products. We add to these criteria whenever we see an opportunity to improve our virus handling. Our compliance with our own criteria is as close as possible to 100%. The criteria are demanding, and we are occasionally forced to drop one or two points for certain viruses where the criteria cannot be met. For instance, the AntiCMOS virus overwrites the MBR. An original MBR cannot be recovered. We have written a better-than-original replacement MBR which obeys all known constraints on content and function, both documented and undocumented.

General anti-virus products must cover thousands of viruses, regardless of their rarity. Many of these products have truly important and exceptional qualities which address virus problems that we do not. We continue to recommend purchasing at least one top-rated anti-virus product and keeping it up-to-date, to use in addition to our product, for better overall coverage.

However, when we compare our score on these 24 criteria with other anti-virus products, we find that we stand apart. This is understandable, given the narrower technical objective we have set for ourselves. We focus strictly on those viruses which are considered epidemic or everyday nuisances.

Our *Lotus* experience has taught us that criteria other than these 24 are far less valuable on a day-to-day basis. Mostly, what our customers demand is immediate and complete relief from epidemic boot sector virus attacks. We think *OverByte* answers this demand well with *DisQuick/ViRemove* diskettes.

WHAT DOES OVERBYTE PRODUCE AND WHAT ARE ITS FEATURES?

We produce pre-formatted diskettes for general sale, and special formats for third-party software companies. These formats and contents are protected under pending copyrights, trademarks, and patents.

DQ™, *DisQuick*™, *ViToler8*™, *ViRemove*™, *DQExpert*™

DisQuick optimizes diskette I/O increasing Read/Write speeds by 30%+ (on a full diskette, 40 seconds is saved on combined Read/Write)

ViToler8 prevents data from begin stored in areas viruses damage (virus tolerance means that the probability of data damage is reduced)

DisQuick launches compliant applications from special boot sectors (we work with other anti-virus companies to launch their products too)

ViRemove detects, identifies, disables, and removes viruses, (we update this to handle more viruses regularly)

DQExpert debugging allows expert user intervention of viruses (we develop new features to allow easy interception of new viruses)

Booting from a *DQ* diskette causes educational text to be displayed. The display program adapts the text to the immediate user needs. Fast use by corporate MIS is easy. Slow comprehensive reading is also.

Our focus on responding to epidemic boot sector viruses at boot-time makes *OverByte* products quite different from other anti-virus programs. Many of our anti-virus operations may be done safely only at boot-time.

Along with *DisQuick/ViRemove*, we have provided a boot-time debugger. *DQExpert* allows the virus professional to examine the PC in detail. New viruses can be analyzed, disabled, and removed before the OS runs. The features of *DQExpert* have been optimized for virus investigation.

WHAT DO WE RECOMMEND TO PREVENT OR WIPE OUT EPIDEMIC BOOT SECTOR VIRUSES?

Identify behaviours that cause virus spread. Educate users to avoid these behaviours. Use all available tools to facilitate this education and behaviour prevention. Determine what features an anti-virus needs to be useful in your organization. Buy the best general anti-virus product with those features. Install it properly and keep it up to date. In addition, we feel that when the majority of diskettes used in an organization have *DQ* technology, the number of virus attacks will decrease dramatically.

OverByte Corporation welcomes licensing to and industrial partnerships with other anti-virus companies.

REFERENCES

[1] Undocumented PC (Frank Van Gilluwe, Addison Wesley)

[2] Undocumented DOS (Andrew Shulman et al, Addison Wesley)

[3] System BIOS for IBM PC/XT/AT Computers and Compatibles (Addison Wesley)

[4] Pentium Processor User's Manual (Intel)

[5] DOS Technical Reference (IBM)

[6] MS-DOS Encyclopedia (Microsoft, Microsoft Press)

[7] PC Interrupts (Ralf Brown & Jim Kyle, Addison Wesley)

[8] The Programmers PC Sourcebook (Thom Hogan, Microsoft Press)

[9] Zen of Assembly Language/Code Optimization (Michael Abrash, Coriolis)

[10] The Waite Group's MS-DOS Developer's Guide (Howard Sams & Company)

ABOUT LOTUS

We will maintain close ties with *Lotus Development Corporation*. Of course, *Lotus* is to be held harmless regarding *OverByte* issues.

Trademarks: *DQ*™, *DisQuick*™, *ViToler8*™, *ViRemove*™, *DQExpert*™

SECURING DOS

Neville Bulsara

Quantum System Software, 52 Regency Chambers, Near Nandi Theatre, Bandra (W), Bombay 400 050, India

Tel +91 22 643 1233 · Fax +91 22 642 2182 · Email neville.bulsara@f1.n606.z6.fidonet.org

ABSTRACT

Ever since the arrival of viruses on the MS/PC-DOS platform, various methods have been adopted to deal with the problem. These methods (virus specific and non-specific) have met with varying degrees of success (or failure!).

What with the increasing number of viruses and the threat posed by the mutation tools looming on the horizon, the problem of glut threatens to swamp the developers of virus specific solutions.

While the importance of scanners cannot be understated, the need for generic solutions increasing seems to be a foregone conclusion. One of the 'weapons' in the armoury of the developers of generic products is the behaviour blocker. Integrity checkers attempt to plug the holes left open by behaviour blockers.

This paper aims then to highlight the fact that existing behaviour blockers and integrity checkers fail to 'SECURE DOS' effectively. It also provides an insight into why we find ourselves in this sorry state!

This paper highlights the author's argument that an effective way to SECURE DOS (as Microsoft seems not to be concerned with the virus menace), would involve parking oneself between the 'D' and the 'OS' of DOS - *that is, intercepting viruses at a level* below *the generic OS calls for file I/O and at a level above the actual physical disk.*

The paper goes on to explain precisely what happens in the 'innards' of DOS with respect to translating file I/O to actual physical disk I/O.

*The paper puts forward a new generic method which the author feels, in conjunction with existing methods of scanning and checksumming, would offer a fair level of robust security. The method would involve the creation of a '**Safe Zone**' (non-writable) on media which would be host to all executable entities. This zone would effectively be a* 'disk within a disk' *(a mountable volume à la DOS's CFV's), which would be mounted via a device driver.*

An insight into how such a device driver would function so as to enable it to determine without user intervention (since the elimination of the associated nuisance of a behaviour blocker in this case is a primary design goal), whether a write to the Safe Zone (read that as creation/modification of an *executable) is legitimate or not etc., is explained in detail. Various methods are discussed to deal with the presence of companion viruses and EXE file header infectors on the Safe Zone.*

The paper ends by highlighting the possible problems which could crop up with such a system in place and how effective such a solution would be under NetWare and Windows.

PREFACE

First things first - **THIS PAPER IS TOO LATE!!** As a matter of fact, **this subject itself is too late.** Securing DOS - at a time when DOS as we (or rather I) understand it is on it's way out? Sure, one may say that Windows '95 will continue to support DOS applications. Yes, it will, but it's still not DOS (as I see it), and hence I repeat - this paper is too late! However, the idea behind it is not.

As we all know, a large number of methodologies exist for combatting the virus menace - Scanners, Behaviour Blockers, Checksummers etc. All of these have their advantages - and their disadvantages. While it is not my intent to debunk any method which one may employ to safeguard their systems, it is only fair that we highlight the major problems associated with the above.

The greatest problem with scanners is that they are able to detect only known viruses. A heuristic scanner may be able to detect an 'unknown' virus, but then there are ways of bypassing such scanners. As someone put it - *'No virus was ever first detected by a scanner'*. The problem with scanners then is that they are **reactive** - someone has to get infected before the scanner is updated to handle a new virus. Nevertheless, because of their irreplaceable propose of detecting known viruses, a scanner is a compulsory weapon in an anti-virus armoury.

The greatest advantage of a checksummer is that it can *'detect unknown viruses'*. As we all know by now, checksummers detect changes, not viruses! Sure, a change may be due to a virus (and then, it may not!). Hence, checksummers are an important tool in generic virus detection. The problem with them, however, is that they are **even more reactive than scanners**. A file must become infected (with a 'known' or 'unknown' virus) before the checksummer does its work. Another major problem with checksummers is that they are liable to be 'led up the garden path' by stealth viruses - unless they use some very low-level routines to bypass the operating system in order to process files.

The only pro-active weapon in the anti-virus armoury is the behaviour blocker. These look out for *'virus-like activity'* rather than viruses. That is to say that they *aim* to 'stop the virus at the letter v' - when it tries to go memory-resident, infect files etc, etc. The problem - or rather the problems - are as follows:

(a) they leave the decision in the hands of the user

(b) they raise false alarms

(c) **they can be bypassed - VERY easily bypassed.**

What, then, is the way out? Do we live with this state of affairs? Or is it time to look at the problem from a different angle - perhaps even redefine the problem? Perhaps the problem can be additionally tackled at a *different level.* Perhaps it's time to redefine the rules of the game!

The problem with trying to solve the problem (no pun intended!) by redefining the rules in the middle of the game is - well, you'll see that for yourselves!

Having been a 'hard-core' DOS programmer (so they tell me) for the last 11+ years and an anti-virus researcher (now THAT, I am!) for the last seven, if 'Securing DOS' is one of the topics of discussion, here then is a *proposed* method of doing so - for better or for worse.

Around half of this paper is devoted to exploring in depth the way DOS handles files. You will find a regular sprinkling of several Data Structures maintained by DOS. At first glance it might seem that this piece is devoted to a dissection of the innards of the DOS filing system, rather than a paper on 'Securing DOS'. ***However, it is not possible to SECURE DOS without knowing what DOS is.*** And the filing system is what DOS is all about (or at least a majority of it is).

I would like to clarify that, in this paper, I'm not suggesting a rigid solution. As a matter of fact, there are no solutions in here. This paper raises more questions than it answers; questions which I've asked myself over the past few years. Some of them I've managed to answer. The rest are for you to ponder. If these questions lead anyone to devise a solution, similar or drastically different from the one I propose, this paper will have served its purpose.

Iacta alea est: The die is cast.

WHEN ARE VIRUSES A PROBLEM?

I became involved with computer viruses in 1988. Back in those days, I used to teach at a computer training institute. They happened to get hit by the Brain virus. It was a bad situation, 60+ infected systems (they used simple PCs - not even XTs - in their training labs back then). All disks were infected, including the student's floppy disks - it wasn't funny! Anyway, it so happened that I was the only person around who could perhaps do anything about it - I did. The rest as they say, is history.

Since then I've served as a consultant to several large organizations, the Government and some Defence Establishments. Over the years, I've come to a conclusion that *viruses (by themselves) are not a problem.* By themselves they're just like any other software or hardware glitch. You're just unlucky if it happens to hit you.

The problem is **THE PROLIFERATION OF A VIRUS.** I've always encouraged organizations that have had a major outbreak to try and reconstruct the sequence of events which led to that outbreak. In over 98% of the cases, it turned out that the virus came in on a single floppy disk. From this entity, it jumps to usually just one system (so far this is NOT a problem you can't cope with). From this system it jumps to other disks and from these to other systems (it could also go over a network). By the time you detect the intruder, scores of systems are infected. *NOW you have a problem on your hands!*

I've always stated that the **earlier** you detect the intruder, the **faster** you can deal with it, saving oneself much heartache later.

The trick then lies in ***DETECTING THE INTRUDER AT THE EARLIEST AND PREVENTING IT FROM PROLIFERATING!***

MY CONCEPT OF SECURING DOS

My concept of securing DOS is securing the executable entities which can be infected by a virus. For the purpose of this paper, I've restricted myself just to securing executable files. Boot Blocks, by comparison, are far easier to secure.

When I talk of securing executable files (before I proceed further and I actually detail how this can be done), it is important that we have a clear understanding of just how DOS deals with files - especially since what we are going to end up doing is securing files.

HOW DOS KEEPS TRACK OF FILES

DOS keeps track of files on a volume using its **file system**. The DOS file system consists of the File Allocation Tables (FATs) and the Root Directory. The Root Directory can contain both files and subdirectories. Each subdirectory can also contain files or further subdirectories.

For the next couple of lines (unless stated otherwise), I will refer to a file as an 'Entity'.

Every entity has an 'entry' (hereafter referred to as a Directory Entry or **DirEntry**) in either the root or a subdirectory. That DirEntry is the starting point from which DOS manages that entity. From the DirEntry, DOS knows (amongst other things) where the file actually starts on the disk, and its length (the file size).

If the DirEntry provides information as to where the file actually starts, the FAT provides information as to where it resides as a whole on the volume. For a detailed understanding of how the DirEntry and FATs are used to keep track of a file, one can refer to several books on the topic.

Suffice it for now to state that the Directory Entries and the FAT are used by DOS to keep track of the actual layout of files on a given medium.

While it is not important for us to know the FAT structure, it is important to know precisely what the DirEntry looks like, as we will be employing it in our scheme. Each Directory Entry is 32 bytes in length and has the following structure:

Table 1 : DirEntry (Directory Entry) Structure

```
DirEntry          STRUC
DE_PrimaryName    db 8 dup (' ')      ; 8 byte primary name
DE_Extension      db 3 dup (' ')      ; 3 byte extension
DE_FileAttrib     db ?                ; 1 byte for file attribute
DE_Reserved       db 10 dup (?)       ; 10 bytes reserved field
DE_FileTime       dw ?                ; 2 bytes (word) file time
DE_FileDate       dw ?                ; 2 bytes (word) file date
DE_Cluster        dw ?                ; 2 bytes (word) starting cluster
DE_Size           dd ?                ; 4 bytes (dword) file size
DirEntry          ENDS
```

As you can see, the DE_Cluster (starting position of the file) and the DE_Size fields in a DirEntry and then using the FAT, DOS is able to get to a required file.

Since we've got around to defining some structures, we might as well define another - especially as we're going to employ it in our proposed scheme. We'll call this structure the **Current Directory Structure (CDS)**. A CDS is maintained for each and every drive in the system. Each CDS Entry **(CDSEntry)** contains the current working directory for that drive (which explains why DOS doesn't forget which directory you're in on drive C when you go to drive A or whatever!). Apart from the current working directory, the CDSEntry contains a pointer to the device driver to be called for performing actual disk I/O on that media; bit attributes which specify the availability of the disk, whether it is JOINed, SUBSTituted, or a network drive, etc.

The CDS is built at boot time by DOS as it processes CONFIG.SYS. If the said file has LASTDRIVE=F, DOS builds CDSEntries for drives A through F. If there were no drives beyond C, the bit attributes for CDSEntries for drives D thru F are masked to indicate that those drives are invalid or not available.

The structure of each entry in the CDS is as follows:

Table 2 : Structure of Entry in CDS (one for each drive till LASTDRIVE)

Structure valid for DOS 4+

```
CDSEntry            STRUC

CDS_CurrentPath     db 67 dup (0)     ; 67 bytes to hold current path for the drive in the
                                      ; form: C:\DOS\VERSION5. The path is null
                                      ; terminated

CDS_DrvAttrib       dw ?              ; 2 bytes that hold flags to indicate whether drive
                                      ; is physical, network JOINed, SUBSTituted etc.

CDS_DPB_Ptr         dd ?              ; far pointer to the Drive Parameter block for this
                                      ; drive

                              For local drives

CDS_StartCluster    dw ?              ; start cluster of current directory

CDS_Unknown1        dd ?              ; unknown

                              For network drives

CDS_Redirector      dd ?              ; far pointer to Redirector

CDS_UserData        dw ?              ; user data from 21h/5f03h

                              For all drives

CDS_SkipCount       dw ?              ; holds count of bytes to skip over when
                                      ; displaying the current directory. Normally 2 so
                                      ; that the drive and the colon are masked.
                                      SUBST and JOIN change this so that only
                                      ; appropriate parts are visible

CDS_Unknown2        db ?              ; unknown

CDS_IFSPtr          dd ?              ; far pointer to IFS driver

CDS_Unknown3        dw ?              ; unknown word

CDSEntry            ENDS
```

Phew! That much information, just so that DOS can remember what the current directory for a drive is? Well, it has some other purposes too, as we shall see below.

INSIDE THE MSDOS FILE SERVICES BY ...

'Inside the IBM PC', by Peter Norton, was the first book which delved into the innards of the PC. A great many programmers got their first taste of low-level activity by reading that book. Before that, it was just a case of:

'That box there does it (though it ain't black!). We don't know how it does it, but it does. So there!'

That's what we're all prone to do. Take things for granted. The classical **'black box' approach.** Supply the (correct) inputs and get the desired output. The black box approach guarantees correct results (we hope!). There's nothing wrong with this approach, but it's taboo to a hacker. A hacker needs to know what makes the box tick. Till he figures that out, well - hopefully you know what I'm talking about!

Let's consider the typical black box approach when it comes to doing things (opening/reading/writing/closing) with a file. If we use the DOS API, all we need to do is (a) Open the file, (b) Read from it, (c) Write to it and (d) Close it. Pure and simple. You call DOS with the correct inputs, DOS does its work. **You're OK, DOS is OK** (with due apologies to the author of I'm OK, you're OK).

But what happens inside DOS when let's say you open C:\COMMAND.COM, read from it, write to it and then close it? Here's what happens in reality:

- DOS indexes into the current PSP to locate a free entry in the **JFT** (Job File Table). If no free entries exist, DOS returns, indicating that there are too many open files.
- Having found a free entry in the JFT, DOS remembers the position of this entry. This index into the JFT eventually will become the **'handle' or the JFN** (Job File Number) which will be returned to your application, assuming that the Open is successful.
- DOS goes through its **SFTs** (System File Tables - more on this later) looking for the first free SFT entry (SFT_Entry). If no SFT entries can be found, DOS returns, indicating an error.
- DOS parses the filename to determine the drive on which the file is supposed to exist (in our example it would be C).
- DOS goes through the **CDS** (remember the CDS?) to determine whether the said drive exists and is valid. If not, DOS returns with an error.
- DOS examines the CDSEntry for that drive to determine whether the drive is ***networked.*** If it is not, DOS uses CDS_DPB_Ptr to derive the address of the Drive Parameter Block (DPB) for that drive. The DPB is used by DOS in order to locate the root directory and the address of the device driver to be called, in order to actually do the low level disk i/o.
- DOS calls the device driver for that drive to read the root directory. It processes what is read to look for COMMAND.COM. If not found, DOS returns with an error code. Remember at this point that what is read is a series of entries of type DirEntry.
- Having found the required file, DOS updates the SFT_RefCount in SFT_Entry. ***The index of this SFT_Entry within the SFT itself is referred to as SFN (System File Number).*** DOS then updates SFT_Entry with information such as the filename, starting cluster, initial open mode, address of the device driver etc.
- The SFN for this file is copied into the index into the JFT (JFN) which DOS had determined at step 2.
- The JFN is returned to the application as the file handle.

Let us assume that the handle returned on the Open request was 'X'. Whenever you wish to refer to this file subsequently, you just call DOS with one of the inputs as X. This is what happens when you try to read from COMMAND.COM:

1. DOS uses X as an index into the JFT to determine the SFN. Having found the SFN, DOS indexes into the SFT in order to locate SFT_Entry for that file.
2. DOS calls the device driver, whose address is stored in the SFT_Entry, to read the block of data.

3. DOS returns any necessary information to the application.

When you try to write to the file, the following happens:

1. Identical to step 1 above
2. DOS examines the initial open mode field stored in that SFT_Entry to determine whether you can write to the file (to prevent writing to files opened in read mode). If not, DOS returns with an error.
3. DOS calls the device driver to write data.
4. DOS returns any necessary information.

When you close the file:

1. You got it! It's identical to step 1 above.
2. DOS calls the device driver to flush any data it might have had in its internal buffers. This includes the updated DirEntry with the new Date/Time stamps, etc.
3. DOS decrements the SFT_RefCount field to indicate that SFT_Entry is now free for use.

All that just to open, read from, write to and close a file? Yep! No one said it was going to be easy!

Since we've been discussing SFTs, we might as well shed some more light on the topic. *An SFT is nothing but a table of SFT_Entries. Each file which is open at any given time under DOS has an entry in the SFT. If Dir_Entry is the structure by which DOS manages a Directory Entry on disk, SFT_Entry is the structure which enables DOS to manage all file-related activities for a given file.* Below is the layout of a System File Table entry:

Table 3 : Structure of System File Table Entry (SFT_Entry)

Structure valid for DOS 4+

SFT_Entry STRUC

SFT_RefCount	dw ?	; count of number of file handles referring to this file
SFT_InitialOpenMode	dw ?	; initial mode (read,write etc) in which the file was ; opened.
SFT_FileAttrib	db ?	; the attribute of the file on disk field is filled up from ; DE_FileAttrib; during the open.
SFT_DeviceInfo	dw ?	; this word holds various bits indicating whether the ; file is remote, the drive number on which the file ; resides, etc.
SFT_Ptr1	dd ?	; far pointer to device driver header or DPB or REDIR ; data
SFT_Cluster	dw ?	; starting cluster for file - field filled up from ; DE_Cluster on Open
SFT_FileTime	dw ?	; file time (taken from DE_FileTime).This field is ; updated whenever the file is written to
SFT_FileDate	dw ?	; file date (taken from DE_FileDate). This field is ; updated whenever the file is written to

```
SFT_Size            dd ?               ; file size - taken from DE_Size on Open. Updated if
                                       ; neccessary when file is written to.

SFT_FilePtr         dd ?               ; file pointer - indicates where the next read or write
                                       ; will occur (relative to start of file)

                              For local files

SFT_RelCluster      dw ?               ; relative cluster within file of last cluster accessed

SFT_Sector          dd ?               ; number of sector containing DirEntry

SFT_SectorIndex     db ?               ; number of DirEntry within sector

                              For remote files

SFT_REDIRIFS_Ptr    dd ?               ; far pointer to REDIRIFS record

SFT_Unknown         db 3 dup (0)       ; unknown

                              For all files

SFT_FCBName         db 11 dup (' ')    ; filename in FCB format

SFT_Share1          db 6 dup (?)       ; information for use by SHARE

SFT_PSP             dw ?               ; PSP address of owner of the file

SFT_Share2          dw ?               ; information for use by SHARE

SFT_AbsCluster      dw ?               ; absolute address of last cluster accessed

SFT_IFSPtr          dd ?               ; far pointer to IFS driver for file

SFT_Entry           ENDS
```

But ***WHAT'S ALL THIS GOT TO DO WITH VIRUSES?***

Well, so far we've made a candle. Now it's time to light it!

WHY DO BEHAVIOUR BLOCKERS FAIL?

As I mentioned before, a behaviour blocker is the only proactive tool which can detect (hope to?) an unknown virus before it manages to infect an entity. Forgetting for the time being the associated nuisance value which comes along with one, you would think that these busters can keep out all viruses - past, present and future. Think again!

You can liken a behaviour blocker to a sentry commissioned to guard the entrances of a room. A behaviour blocker posts sentinels at every entrance (read that as **KNOWN ENTRANCES**) to the room in order to screen visitors for suspicious objects. So you post one at those two doors and one each at the five windows. Then pray like hell!

The problem is ***you didn't build that room.*** Neither did the security agency (read that as anti-virus vendor) which supplied the guards. The room (as a matter of fact the whole building) was built by that company up in Redmond. And for all you know, that room of yours could have a **trapdoor** that neither you nor the Vendor know anything about. Did I say ONE? Well, think perhaps in the hundreds! Anyway, that's the loophole an intruder could exploit to get into your system - and they do.

Let us consider a few methods used by some of the more 'novel' viruses to bypass behaviour blockers:

- Opening a file in Read mode (this allows the behaviour blocker to allow the request). Upon return from DOS, use the handle to index into the JFT; from there get the corresponding SFT_Entry; twiddle with the SFT_InitialOpenMode field to indicate that file opened in Read+Write mode.
- Use the starting cluster field in the DirEntry; process the FAT; infect the file using Int 25h/26h, thereby bypassing any Int 21h handlers set up by a behaviour blocker.
- Use the start cluster field to infect the header of EXE files. Again using Int 25h/26h bypass Int 21h.
- Using Int 13h to monitor reads to disks and 'infecting' any sector which starts with the 'MZ' indicator.
- Twiddling with the starting cluster field in the SFT_Entry in order to 'open a file, but write to another'.
- Using various tunneling techniques to derive the address of the original interrupt handlers and subsequently making far calls to them.
- Using 'reserved' interrupts/functions in order to open, read from,write to files (e.g. using the undocumented DOS Indirect Server Call).
- Infecting the file system itself - DIR-II, for example.
- Modifying the contents of DOS's buffers (infecting them), then setting their bits to indicate that the buffer is dirty, causing DOS itself to flush their contents to disk.

Get the idea? Too many **unguarded** DOS, WINDOWS, and safety NetWares (sorry that should read doors, windows and nets), about which one knows little or nothing.

THE PREREQUISITES FOR WINNING A BATTLE

Sun Tzu, the author of 'The Art of War' wrote as follows:

'Do not understand yourself? You will lose 100 percent of the time. Understand yourself? You will lose 50 percent of the time. Understand yourself and your opponent? You will win 100% of the time.'

Well we so far have understood ourselves (the DOS filing system). We understand - or at least try to - the methods cooked up by the opponents. Unfortunately, the game being as it is, it is impossible to win all the time. But at least we can try.

By now I have hopefully lit a candle or two. It is now time to feed the flame.

SECURING AN OS

As I view it, the way to secure any OS from viruses is really very simple. If a single word could describe it, the word that fits the bill would be **SEGREGATION**.

The current problem (to the best of my limited knowledge) lies in the fact that all ***Operating Systems store both data and program files on the same medium***. That is to say, both these types of files lie on the same volumes, and the OS uses the same filing system and the same file system (File system = FATs, Directories etc. Filing system = routines to manage the file system) to keep track of them. It is hence obvious that *if the OS fails to distinguish between code and data, viruses will exploit this loophole and continue to proliferate.*

The key then lies in SEGREGATING data files and program files; storing them on 'different volumes', and using different file systems and filing systems to manage them. If this segregation can be achieved, then program files can reside on a so called 'Safe Zone' - to coin a new term. This **Safe Zone** (hereafter referred to as SZ) would be a read-only volume managed exclusively by the OS. Since the SZ would be read-only, a new virus introduced into the system would be unable to spread (though it would attempt to). However, the OS could keep track of these write requests, and maintain a log. Frequent examination of this log would display strange behaviour and would indicate the fact that a possible virus is attempting to spread - albeit unsuccessfully. Presto - you have just secured your OS!

Seems simple enough - but there is a lot more to it, as we shall shortly see.

SECURING DOS USING THE SEGREGATION PRINCIPLE

Since DOS (as it stands today) has no concept of SEGREGATION, it is obvious that such a concept would need to be externally induced. I have been thinking about this concept for the last two plus years. I have conducted various experiments which have shown that it is TECHNICALLY feasible to employ such a scheme.

The seed for this idea was sown shortly after I disassembled the infamous DIR-II virus to see what made it tick. After studying the virus, I reached a conclusion that the only way to secure DOS pro-actively was by parking yourself between the D and the OS of DOS. That is to say, at a level between the OS and the actual physical media. The proposed method does precisely that.

It is said that examples bring the obscure to life. Keeping this in mind, we'll proceed with a real example of how such a system is implemented.

Let us consider a system having a single partition (drive C). As things stand, the following directories and files exist on that volume:

C:\		(root directory)
	IO.SYS	(executable entity)
	MSDOS.SYS	(executable entity)
	COMMAND.COM	(executable entity)
	CONFIG.SYS	(executable entity)
	AUTOEXEC.BAT	(executable entity)
C:\DOS		(sub directory)
	HIMEM.SYS	(executable entity)
	EMM386.EXE	(executable entity)
C:\NC		(sub directory)
	NC.EXE	(executable entity)
	NCMAIN.EXE	(executable entity)
	NC.INI	(non-executable entity)
	NC.MNU	(non-executable entity)

Our objective at this stage is the creation of a Safe Zone (SZ) which will be host to executable entities. All non-executables will reside on a normal volume called the Data Zone (DZ).

STAGE 1 - PREPARATION OF THE SZ

A program (let's call it SZPREP) is run. **SZPREP** sweeps through your hard disk determining the directory tree and collecting the names of all executable entities which can be moved to the SZ. It determines that C:\COMMAND.COM, C:\AUTOEXEC.BAT, C:\NC\NC.EXE and C:\NC\NCMAIN.EXE can be moved to the SZ. (IO.SYS, MSDOS.SYS, CONFIG.SYS, HIMEM and EMM386 cannot be moved, as they must load prior to any other drivers).

Having determined the names (and the total size) of the files which can be moved to the SZ, SZPREP creates *a hidden read-only file in the root directory of drive C*. Let us call this file SAFEZONE.NB (how imaginative!). A more generic term for this file would be a ***Safe File Volume (SFV)*** - there, I've just coined another new term!

We need to clarify at this point itself what the structure of the SFV will be like. Since it will essentially play host to files, it must have its own file system and data space. The exact structure of the file system is a matter of design. For simplicity, it can be almost identical to the DOS file system - i.e. it can have a boot sector, FAT, directory and data area. I would suggest that, instead of having the concept of sub directories, a flat directory structure, using the 10 unused bytes in DirEntry (DE_Reserved) to store the checksum of the path where this file resides (refer to Table 1) be employed.

SZPREP then moves (copy to destination, delete from source) all the above-mentioned files which are candidates for being moved into the SFV. SZPREP then plonks a file (let's call it **SZMOUNT.SYS**) into the root directory of drive C, and modifies CONFIG.SYS. Your SZ is now ready, and drive C looks like:

C:\		(Root Directory)
	IO.SYS	(executable entity - can't be moved)
	MSDOS.SYS	(executable entity - can't be moved)
	CONFIG.SYS	(executable entity - can't be moved)
	SAFEZONE.NB	**(SFV)**
	SZMOUNT.SYS	**(executable entity to manage the SFV)**
C:\DOS		(Sub directory)
	HIMEM.SYS	(executable entity - can't be moved)
	EMM386.EXE	(executable entity - can't be moved)
C:\NC		(Sub Directory)
	NC.INI	(non-executable entity)
	NC.MNU	(non-executable entity)

As we can observe, all but the barest minimum of infectable objects have 'disappeared' from the volume. In a real-world situation, hundreds of objects would have been moved into the SZ, leaving just six possible candidates for infection.

STAGE 2 - MOUNTING THE SZ

The SZ is mounted at the time when the system is booted off the hard disk. DOS processes CONFIG.SYS, and loads the SZMOUNT.SYS driver. SZMOUNT detects the presence of the SFV, reads it (in order to determine its size, volume characteristics) and then (as one would guess) implements it as a drive - à la STACKER, DBLSPACE, DRVSPACE - right?

WRONG!

In the case of the above mentioned 'drive doublers', they would either treat the SFV (in their case it is a CFV - Compressed File Volume) as C: and the original drive as D: - or vice-versa. Our software can't afford to do that, as most programs expect their non-executable components (configuration files, INI files etc) to reside in the same drive and directory as they do. As for example, NC.EXE would expect NC.INI and NC.MNU to be present where it loaded. Hence, NC would not work (as would not a large number of other programs) if we implemented our SZ as C and the original volume as drive D or vice-versa.

What we need to do is make it seem as though they exist on the same drive, which in reality they do not. This calls for a bit of magic!

Keep in mind too that we need to monitor all activities (such as DIR, Open, Read, Write, Close, Delete, MD, RD, CD, Get Disk free space etc) - that are in any way connected with disk I/O. We need to pass these requests seamlessly either to the original volume or to the SZ. And most important, both volumes must look like the same original volume. And in case we've forgotten, we've got to maintain the integrity of files on the SZ.

In a nutshell, here's what SZMOUNT does when it loads up:

- copy the CDS Entry (refer to Table 2) for drive C: (The DZ) into one of its internal buffers
- take over the **DOS interrupt 21h** (practically all the file/directory handling calls require an Int 21h)
- take over the **DOS Multiplex interrupt (Int 2Fh)**
- determine the address of the DOS Swappable Data Area (SDA - more on this later).

That is all that SZMOUNT needs to do when it loads up. The rest, which is the integration of the SZ with the DZ, is sheer magic!!!

STAGE 3 - SEAMLESS INTEGRATION WITH THE DZ

Stage 3 comes into play the moment Stage 2 is over. It works as follows:

On the occurrence of any Int 21h, control comes to the Int 21h handler inside SZMount. Again, as it is far simpler to explain using examples, I shall supply a few.

Let us suppose that SYSINIT (part of the DOS startup-code) finished processing CONFIG.SYS. It then needed to allocate some memory. SYSINIT issues an Int 21h, with the AH register = 48h. This is trapped by SZMOUNT's Int 21h handler. SZMOUNT goes through a lookup table, and determines that ***this function does not deserve its attention, and so LOWERS a flag, and passes control to DOS to do the necessary.***

On the other hand, let us suppose that the request was to **open** a file (let us say C:\NC\NC.MNU). SZMOUNT determines that an open file request is ***one of the many requests which is deserving of its attention. So, it RAISES a flag and passes control to DOS.***

If we recollect our earlier discussion 'Inside the MS-DOS File System', DOS essentially checks the CDS, determines that C: is a valid drive, calls the device driver associated with the drive, fills up a SFTEntry etc,etc, and then returns with a handle (JFN) specifying that the open was successful.

At this stage, control comes back to SZMOUNT before it returns to the application (as DOS was called by SZMOUNT which was called by the Application). At this point, SZMOUNT ***checks whether its flag is in a raised state (it is).*** Now, SZMOUNT ***checks whether DOS has returned an error (it hasn't).*** So, SZMOUNT LOWERS its flag and returns to the application. This is how SZMOUNT manages file opens on the DZ (actually, DOS does). As a matter of fact, this is how SZMOUNT manages any file/directory related activities on the DZ.

What then, if an application wanted to ***open C:\NC\NC.EXE. Well, the flag is RAISED, control passes down to DOS but DOS can't find an NC.EXE, and so returns to SZMOUNT with an error. SZMOUNT sees that its flag is raised (a raised flag indicates POSSIBLE activity may be required on the SZ) AND DOS has returned an error. This tells SZMOUNT that it should attempt to repeat that activity on the SZ.*** And at this point the Int 21h handler of SZMOUNT begins its real juggling act.

First, the Int 21h handler ***twiddles with the CDS entry for drive C, indicating that the drive is a NETWORK drive.***

Second, the handler refers to the DOS SDA to find out the contents of the various registers when DOS's Int 21h handler got control of the request (the contents of registers, along with other information is stored within the SDA). The registers are reloaded with the stored contents.

Lastly, SZMOUNT lowers its flag and REISSUES the call to DOS!

This time across, DOS examines the CDS for drive C, and discovers that it's a NETWORK (remote) drive. A NETWORK drive, as far as DOS knows, may not have a DOS compatible FAT, and can be implemented by any vendor he/she deems fit.

In order to integrate alien not-FAT file systems with DOS, DOS provides hooks by which it calls programs to implement remote drives. These hooks constitute the DOS ***Network Redirector Interface***, and a program using these hooks is called the Network Redirector. The Network Redirector interface is called via Int 2Fh (that is why SZMOUNT captured that interrupt, too) with the AH register loaded with the value 11h, and the AL register loaded with the sub-function (Open, Read, Write, Close etc.)

To cut a long story short, DOS, finding that the drive is a network drive, issues an Int 2Fh with AH=11h and AL=16h (open existing file). DOS also passes a pointer to an uninitialized SFT along with other relevant information in the SDA.

On receiving control via Int 2Fh, SZMOUNT determines the function, derives a pointer to the filename (C:\NC\NC.EXE), reads the file system from the SZ into memory (at the disk level, not making any DOS calls!), and ending the file, fills information into the supplied SFT. At this point, SZMOUNT twiddles with the SFT entry for that file to indicate that this file is a remote file. SZMOUNT then twiddles with the CDS for drive C, to indicate it is a physical drive, and returns. Application wanted the file opened; SZMOUNT (and not DOS) has opened it for you.

Similarly, if you issued a 'DIR C:\NC' command, first the files on the DZ followed by the files on the SZ would be displayed - despite the fact that they lie on seperate volumes! Well, actually, a DIR (that translates into a series of find first/find next calls to the OS) requires a bit of special handling (as do the remove directory and the rename functions), but it is possible to do it.

Reads (and writes) do not raise the flag in the Int 21h handler of SZMount. The handler does not need to meddle with the CDS to indicate that the drive is networked. A read request on a file will result in DOS issuing an Int 2Fh, AH=11h, AL=8 if the SFT for that file indicates that the file is remote. Hence, if a file on the SZ needs to be read from (or written to), the Redirector gets control and does its job (actually reading but never writing).

Similarly, various functions to take care of setting attributes, deleting files, managing directories, etc, can be seamlessly integrated as ***DOS will always call the Redirector when it needs to access 'networked drives'.***