Internal distribution code:

(A) [-] Publication in OJ
(B) [-] To Chairmen and Members
(C) [-] To Chairmen
(D) [X] No distribution

Datasheet for the decision of 21 February 2017

Case Number: T 1573/11 - 3.5.06

Application Number: 06700770.8

Publication Number: 1842148

IPC: G06F21/00, G06F21/22

Language of the proceedings: EN

Title of invention:
COMPUTER PROTECTION AGAINST MALWARE AFFECTION

Applicant:
William Grant Rothwell

Headword:
Malware protection/ROTHWELL

Relevant legal provisions:
EPC 1973 Art. 56

Keyword:
Inventive step - (no)

Decisions cited:

Catchword:

Europäisches Patentamt
European Patent Office
Office européen des brevets

Beschwerdekammern
Boards of Appeal
Chambres de recours

European Patent Office
D-80298 MUNICH
GERMANY
Tel. +49 (0) 89 2399-0
Fax +49 (0) 89 2399-4465

Case Number: T 1573/11 - 3.5.06

D E C I S I O N
of Technical Board of Appeal 3.5.06
of 21 February 2017

Appellant: (Applicant)	William Grant Rothwell Lehnmattstrasse 3 4573 Lohn-Ammannsegg (CH)
Representative:	Suckling, Andrew Michael Marks & Clerk LLP Fletcher House Heatley Road The Oxford Science Park Oxford OX4 4GE (GB)
Decision under appeal:	**Decision of the Examining Division of the European Patent Office posted on 23 February 2011 refusing European patent application No. 06700770.8 pursuant to Article 97(2) EPC.**

Composition of the Board:

Chairman	W. Sekretaruk
Members:	M. Müller
	G. Zucka

Summary of Facts and Submissions

I. The appeal lies against the decision of the examining division, with reasons dispatched on 23 February 2011, to refuse the application for lack of inventive step over document

D5: Miretskiy Y *et al.*, "Avfs: An On-Access Anti-Virus File System", preprint retrieved from the Internet, marked "appears in the proceedings of the 13th USENIX Security Symposium (Security 2004)".

The symposium was held from 9 to 13 August 2004 in San Diego, CA, and the preprint duly appeared in the symposium proceedings published in 2004 by the USENIX association.

The decision also cites further documents without relying on them in its reasons, including

D1: Schmid M *et al.*, "Protecting data from malicious software", Proceedings of the 18th Annual Computer Security Applications Conference (ACSAC'02), pages 199-208, IEEE Press, 2002, and
D2: Russinovich M, "Inside On-Access Virus Scanners", Windows & .NET Magazine Network, InstantDoc #300, 1997.

II. Notice of appeal was filed on 4 May 2011, the appeal fee being paid on the same day. A statement of grounds of appeal was received on 29 June 2011. The appellant requested that the decision be set aside and that a patent be granted on the basis of claims according to a main or one of three auxiliary requests as filed with the grounds of appeal. Auxiliary request 1 comprises claims 1-25, the other three requests comprise claims 1-27. The further documents on file are drawing

sheets 1-5 as published, and description pages 1, 2, 9-22, 24-38 as published, pages 5, 6, 23 as annexed to the International Preliminary Examination Report, page 4 as filed on entry into the regional phase, and pages 3, 7, 7a, and 8 as filed on 24 November 2009.

III. In an annex to a summons to oral proceedings, the board informed the appellant of its preliminary opinion that the claimed invention according to all requests lacked inventive step over D5 and common general knowledge in the art. Some clarity objections were also raised.

IV. In response to the summons, with letter of 20 January 2017, the appellant filed amended claims 1-27 according to fourth and fifth auxiliary requests and claims 1-26 according to a sixth auxiliary request.

V. During the oral proceedings, the appellant requested that claim 1 of the fourth and fifth auxiliary requests be amended by incorporating the additional features of claim 19.

VI. Claim 1 of the *main request* reads as follows:

"A method of thwarting malware at its propagation phase to protect a computer (1) against malware affection, the computer (1) having a data storage (157, 12, 2) and an operating system for managing the data storage (157, 12, 2), the method comprising providing a filter module in the operating system which operates to detect an attempt to store data in the data storage (157, 12, 2), to determine a data format of the data to be stored in the data storage (157, 12, 2), and to prevent storage of the data if the data format is determined to relate to a predefined type, characterised in that the filter

module operates to prevent storage of the data without user confirmation, and in that the predefined type of data format is an executable data format, thereby blocking an unauthorised attempt to write data to the data storage (157, 12, 2) that could potentially constitute malware, and thereby preventing malware from propagating by preventing it from saving its execution code to the data storage (157, 12, 2)."

VII. Claim 1 of the *first auxiliary request* is identical to that of the main request, except that before the clause beginning with "thereby" the following phrase is added:

"... and in that the format of the data to be stored is determined by examining the intended file extension of the file to be created or by examining a file header of the file to be written, ...".

During the oral proceedings the appellant indicated its willingness to limit the claim to the second ("file header") alternative.

VIII. Claim 1 of the *second auxiliary request* reads as follows (differences with claim 1 of the main request are underlined or struck through):

"A method of thwarting malware at its propagation phase to protect a computer (1) against malware affection, the computer (1) having a data storage (157, 12, 2) and an operating system for managing the data storage (157, 12, 2), the method comprising providing a filter module in the operating system which operates to detect an attempt to store data in the data storage (157, 12, 2), to determine whether a data format of the data to be stored in the data storage (157, 12, 2) is an executable data format or a non-executable data format,

and to prevent storage of the data without user confirmation if the data is determined to be an executable data format ~~the data format is determined to relate to a predefined type, characterised in that the filter module operates to prevent storage of the data without user confirmation, and in that the predefined type of data format is an executable data format~~, thereby blocking an unauthorised attempt to write data to the data storage (157, 12, 2) that could potentially constitute malware, and thereby preventing malware from propagating by preventing it from saving its execution code to the data storage (157, 12, 2)."

IX. Claim 1 of the *third auxiliary request* is identical to that of the second auxiliary request, except that the determination of whether the data format is an executable or a non-executable data format is specified to occur

"... by examining a file extension or a file header for the data ...".

X. Claim 1 of the *fourth auxiliary request* is identical to claim 1 of the main request, except that the phrase "without user confirmation" is replaced by "without prompting the user for confirmation", and that it is specified that the filter module in the operating system

"... operates, in a first mode, to detect an attempt to store data in the data storage (157, 12, 2), ..."

Claim 19 of the fourth auxiliary request reads as follows:

"A method as claimed in any preceding claim, comprising providing an administration application to enable the user to run the filter module in a second mode so as to turn off the blocking, for example in order to allow the user to install software or perform other routine maintenance."

XI. Claim 1 of the *fifth auxiliary request* is based on that of the second auxiliary request and reads as follows (differences underlined or struck through):

"A method of thwarting malware at its propagation phase to protect a computer (1) against malware affection, the computer (1) having a data storage (157, 12, 2) and an operating system for managing the data storage (157, 12, 2), the method comprising providing a filter module in the operating system which, in a first mode operates to

detect an attempt to store data in the data storage (157, 12, 2), ~~to~~

determine whether the data to be stored in the data storage (157, 12, 2) is an executable data format or a non-executable data format,

if the data to be stored in the data storage (157, 12, 2) is an executable data format, determine whether the request is from a process pre-defined by a user as a trusted process or is made under a system account pre-defined by the user as a trusted system account, and ~~to~~

prevent storage of the data without requiring user confirmation ~~if~~ whenever the data is determined to be an executable data format and the request is not determined to be from a process pre-defined by the user as a trusted process or made under a system account pre-defined by the user as a trusted system account, otherwise allow storage of the data, thereby blocking

an unauthorised attempt to write data to the data storage (157, 12, 2) that could potentially constitute malware, and thereby preventing malware from propagating by preventing it from saving its execution code to the data storage (157, 12, 2)."

Claim 19 of the fifth auxiliary request reads as follows:

"A method as claimed in any preceding claim, comprising providing an administration application to enable the user to, in a second mode, turn off the blocking, for example in order to allow the user to install software or perform other routine maintenance."

XII. Claim 1 of the *sixth auxiliary request* is based on that of the third auxiliary request and reads as follows (differences underlined or struck through):

"A method of thwarting malware at its propagation phase to protect a computer (1) against malware affection, the computer (1) having a data storage (157, 12, 2) and an operating system for managing the data storage (157, 12, 2), the method comprising providing a file system filter module (153) in a kernel (15) of ~~in~~ the operating system which operates to detect an attempt to store data in the data storage (157, 12, 2) , to determine whether the data to be stored in the data storage (157, 12, 2) is an executable data format or a non-executable data format by intercepting and examining ~~a file extension or~~ a write data request associated with the attempt to store data in the data storage for a file header for the data, and to prevent storage of the data without user confirmation if the data is determined to be an executable data format, thereby blocking an unauthorised attempt to write data

to the data storage (157, 12, 2) that could potentially constitute malware, and thereby preventing malware from propagating by preventing it from saving its execution code to the data storage (157, 12, 2)."

XIII. All requests also comprise an independent computer claim corresponding largely to the independent method claims reproduced above.

XIV. At the end of the oral proceedings, the chairman announced the board's decision.

Reasons for the Decision

Article 11 RBPA

1. The decision under appeal (see reasons 16) characterises the difference between then claim 1 and D5 as "a particular implementation of a non-technical policy to prevent storing of executable data". The policy being non-technical, it is considered to be given to the skilled person "as part of a requirements specification", and its implementation is found to be obvious. In the minutes of the oral proceedings (see page 3, third paragraph from the bottom) this approach is stated to follow "recommended practice within the EPO" following T 641/00.

2. The appellant complains that the argument of the examining division based on T 641/00 was raised for the first time during the oral proceedings and, thus, came as a "complete surprise" (see grounds of appeal, e.g. points 5 and 8). The examining division's justification that a similar objection had been raised before, albeit

referring to a "generic policy" rather than a "non-technical" one (see minutes of the oral proceedings, page 1, bullet points and enclosing paragraphs) is dismissed because the two arguments were clearly distinct (grounds of appeal, point 9). Reimbursement of the appeal fee is not requested.

3. The board accepts that the argument that a generic policy cannot involve an inventive step is not identical to the argument that a policy is non-technical and hence does not contribute to inventive step. To that extent, the examining division does indeed appear to have presented a new argument during oral proceedings.

3.1 However, the board disagrees that the new argument is significantly different from the earlier one. More specifically, the board disagrees that the new consideration, namely that the claimed policy may be non-technical, even if surprising to the applicant, was so complex that it could not be dealt with during the oral proceedings before the examining division.

3.2 The board therefore considers that the behaviour of the examining division does not show a fundamental deficiency in the sense of Article 11 RPBA or a substantial procedural violation in the sense of Rule 103(1)(a) EPC.

3.3 In passing the board notes that an immediate remittal to the examining division under Article 11 RPBA would not have served any purpose, because the decision, in a section entitled "Further comments" (points 18-20), also indicates that and why the claimed invention was obvious even assuming that the "requirement to deny

storing executable content was considered a technical feature".

3.4 The board therefore decided not to remit the case to the examining division pursuant to Article 11 RPBA without assessing its merits.

The invention

4. The application relates to protecting a computer against malware, where "malware" is understood to comprise all kinds of "computer programs performing actions on computer systems without the consent of a user, often developed for the purpose of doing harm to the computer system or the user" (page 1, paragraph 2).

4.1 A typical prior-art solution is to match suspicious data against malware descriptions. However, this solution can protect computers against malware only once its description is provided. Hence, its effectiveness depends on the "up-to-dateness" of the malware descriptions and is limited to *known* malware (see paragraph bridging pages 1 and 2). Apart from "pattern scanning", the description mentions further prior-art solutions for detecting malware and briefly discusses their disadvantages (see page 18, lines 4-22).

4.2 The application proposes to intercept writing attempts at the file system level, for example the Windows file system command IRP_MJ_CREATE, which is called whenever a new file is created or an existing file is modified (see e.g. page 23, penultimate paragraph, and page 24, paragraph 4). Since, as is observed, "From the operating system perspective, malware is just another form of application program" (page 17, paragraph 4), it is further proposed to block the writing of all

executable files unless the user has given his explicit approval, e.g. when installing new software (page 17, paragraphs 3 and 5). This is referred to as a "deny-write-by-default policy" (*loc. cit.*). The user can approve the storing of executable files for individual "trusted" processes or user accounts, or switch off the protection mechanism entirely (see page 28, last paragraph, to page 29, paragraph 2).

4.3 It is disclosed that executables are detected by their file extension (see page 11, lines 5-16), a known approach (*loc. cit.* and page 24, lines 4-7), or by inspection of the file content to address the possibility that malware might be hidden behind an unexpected file extension (page 24, lines 24-27).

4.4 The application focuses on an implementation of the invention in the Windows operating system (see in particular page 22, paragraph 3, *et seq.*) but stresses that the invention is not intended to limited to that (page 36, last paragraph, to page 38, paragraph 2).

The prior art

5. D1, cited in the application on page 2 (paragraph 2), discloses a system introducing two dedicated "file system permissions": *confirm_on_read* and *confirm_on_write*. A file with these "permissions" cannot be read or written, respectively, without user confirmation (see sections 2, 3.1 and 3.2, and table 1). D1 discloses the possibility to give the same file permission to all files of a given type, including executables (see table 1, ".EXE"), and in which additional rules are provided (e.g. by the users) in order to reduce the number of false alarms (see section 3.4).

6. D2 discloses an implementation of on-access virus scanning for Windows NT based on a "file system filter driver" intercepting file system requests in order to pass them on, reject them or modify them (see page 1, paragraphs 4 and 7, and page 2, paragraph 3). The intercepted requests can be create, read or write (see page 1, paragraph 5, and page 2, paragraph 3). D2 further discloses the possibility of limiting the scanning to certain types of files as defined by the file extension (page 2, still paragraph 3). When a virus is detected, access to the file in question is denied. It is also disclosed that the file may be stored in a specific location, deleted or repaired (see page 2, paragraphs 6 and 7).

7. D5 discloses an "on-access file system" Avfs that prevents malicious data from being "committed to disk" (abstract and paragraph 2). Avfs is located "in the file system" and scans data before it is written to disk (page 3, left column, and figure 1). In principle, viruses are detected by scanning suspicious files for known patterns (for this, the available virus scanner ClamAV is used). It is disclosed that virus scanning may be limited to suspicious files, identified e.g. by executable file type or based on information in "certain regions of files" (see section 5, lines 12-19). D5 also makes a point of not involving the user (see section 2, under "Transparent").

Claim construction

8. Claim 1 of all requests is for a "method of thwarting malware at its propagation phase". According to the description (see page 1), the "propagation phase" comprises in particular the storing of malware at a

target computer, and "thwarting" is substantially synonymous to "preventing". Malware being, for the operating system, "just another form of application program" (see page 17, paragraph 4), the board takes this phrase to mean "method of preventing the storage of executables". The appellant did not challenge this interpretation in the oral proceedings.

9. The other issues raised in the summons regarding clarity and construction of the claims are immaterial for this decision.

Inventive step

10. In the board's view, any of documents D1, D2 and D5 are suitable starting points for the assessment of inventive step of claim 1. The board however prefers to start that assessment by considering common knowledge in the art.

11. It is undisputed that malware scanners of the type discussed in D2 and D5 and as summarised in the application (paragraph bridging pages 1 and 2) were commonly known in the art.

11.1 Fundamentally, a malware scanner tries to establish - or, at least, produce positive evidence - that a file contains malware. If such evidence is found, protective action is taken. If not, the scanned file is deemed free of malware.

11.2 With this approach it is inevitable, that not all malware is captured. In particular, yet unknown ("zero-day") malware cannot be detected.

12. The board considers that the relevant skilled person in the case to hand is the person responsible for designing and deploying a security architecture. Security architects routinely assess whether the security measures in place satisfy the security requirements. It is noted in passing that the required level of security is, in the board's view, part of the problem to be solved by a security architect and not, in itself, part of the solution. A security architect knows that external developments may affect the sufficiency of a security architecture and will therefore have taken into account the increased threat by malware mentioned by the appellant (see e.g. the grounds of appeal, page 4, penultimate paragraph, to page 5, paragraph 3). In general, security architects must strike a balance between cost and benefit when designing or deploying a security measure. In the context of malware protection this is in particular the balance between the inconvenience caused by a security measure and the risk of overlooking malware. In a security-critical context, preference will be given to higher security even at the cost of less convenience for the users; in a less security-critical context a security gap may be tolerable and outweighed by increased user convenience.

12.1 In view of this, the board takes the view that the security architect would have been aware of the mentioned inevitable residual risk that a malware scanner may overlook a piece of malware, and would have assessed whether it was tolerable in view of the given security requirements.

12.2 Moreover, the board considers it obvious that a security architect would, in an individual case, have

come to the conclusion that a risk of overlooking malware may not, or no longer, be tolerable.

13. The appellant argued that the security architects would, in such a situation, have tried to improve the conventional malware scanners, for instance by combining malware scanning with anomaly detection for those files in which no malware was detected, and would not have considered taking the radically different approach according to the invention.

13.1 The board agrees with the first part of the appellant's assertion but disagrees with the second. That is, the board takes the view that the security architect would be aware if no available security architecture could provide the required level of security.

13.2 In particular, the security architect would, if considering that the residual risk of overlooking malware in all available systems was (or had become) intolerably large, not hesitate to consider alternatives.

13.3 In other words, the problem of seeking such alternatives would, in the board's conclusion, realistically have arisen before the priority date.

14. The invention proposes, as a solution to the cited security problem, to prevent storage of any executable code.

14.1 The board notes that this policy does not improve malware scanning - in the sense of achieving its purpose in a better or faster way - but achieves a different goal. The default in a malware scanner is to accept a file unless its infection is established; the invention proposes by default not to accept any

(executable) file unless user approval can be obtained. The board also notes that the invention does not make malware scanning superfluous: whenever an exception to the strict policy is made (as must be done at least for the installation of new software), the problem of making sure that stored executables are not infected reappears. The invention thus cannot replace the known malware scanners entirely but only for end-users; system administrators might still need them.

14.2 Given that the skilled person knows that malware is generally executable code (as stated in the application itself but also implied by D1 and D5; see the above summaries in points 4, 5 and 7), it would have been, in the board's view, an obvious option to consider the very restrictive and conservative security policy, e.g. as a last resort in the absence of any less intrusive but equally secure solutions, to prevent end users from storing any executable code at all.

15. At this juncture the appellant argued that it was an indication of inventive step that, according to the documents on file, the strict policy underlying the present invention had not been considered in the art before, in particular since the malware problem was a well-known one (see also the appellant's letter of 29 June 2011, page 8, paragraph 3).

15.1 The board disagrees. Whether or not an invention was or was not considered in the prior art is primarily an issue of novelty.

15.2 While the board does not want to rule out the possibility that it may be an indication of inventive step if an application satisfies a "long-felt need" in producing a solution to a pressing problem that has

been known for a long time but for which no solution has yet been found, it is of the opinion that this is not the case here.

15.3 As explained above, the skilled person is aware that the security assessment of a given solution may change over time in view of the external circumstances. Specifically, the residual risk of a malware scanner may become intolerable with the rising frequency of new malware. If circumstances change, the skilled person's motivation to address a certain problem may change even if the solution itself is, and always has been, straightforward from a technical point of view.

Main request

16. Beyond rephrasing the policy which, as argued above, the board considers to be obvious *per se*, claim 1 specifies the use of a "filter module in the operating system which operates to detect an attempt to store data in the data storage", determining whether the "data format" is an "executable data format" and, if so, preventing the storage.

16.1 The skilled person, trying to implement said policy, would, in the board's view, come across D2 and D5, each of which discloses the interception of write-access requests by a component that constitutes a "filter module in the operating system" as claimed.

16.2 If, as a matter of policy, executables must not be stored, they must be identified beforehand. The board considers it common place to identify an executable "data format" based on the file extension.

16.3 Not asking a user before making the decision to prevent the storage of a file is an obvious matter of security, because it avoids the risk of that user giving his approval based on the wrong assumption that a file is malware-free.

16.4 The board thus comes to the conclusion that claim 1 of the main request lacks inventive step over common general knowledge in the art in combination with D2 or D5, Article 56 EPC 1973.

First and third auxiliary requests

17. As just stated, determining that a file has an "executable data format" based on the file extension is a well-known and obvious approach. This covers the first alternative of the feature added to claim 1 of the first and third auxiliary requests.

18. The other alternative is to detect an executable data format by inspecting the file header.

18.1 The board takes the view that the skilled person knows that file extensions can be freely changed and thus do not necessarily correspond to the file content. It is thus, in the board's view, an obvious matter that an executable file may "hide behind" an non-suspect file extension.

18.2 It is thus obvious for the skilled person to consider the file content in order to determine whether the file is an executable. In doing so, checking the file header is one immediately obvious option for the skilled person.

18.3 In summary, neither alternative in claim 1 of the first and third auxiliary requests is sufficient to establish an inventive step over the main request, Article 56 EPC 1973.

Second and sixth auxiliary requests

19. The board considers that the amendments to claim 1 of the second auxiliary request are merely clarifications which do not affect the above inventive-step assessment. This opinion was expressed both in the annex to the summons to oral proceedings (point 19) and during the oral proceedings, and the appellant has not challenged it. With respect to the express specification in claim 1 of the sixth auxiliary request that the filter module is a "system filter module in a kernel of the operating system", the board notes that at least D2 expressly discloses the filter module to be in "the kernel" (see figure 2). The appellant agreed during oral proceedings that the filter-module feature did not constitute a feature that was crucial for inventive step. Claim 1 of the second and sixth auxiliary requests thus also lacks inventive step, Article 56 EPC 1973.

Fourth auxiliary request

20. The amendments to claim 1 of the first auxiliary request (including, as stated above, the feature of claim 19) essentially indicate that the prohibition of storing executable files can be switched off. The board takes the view that switching off a security measure is an obvious option in exceptional situations - irrespective of the obvious fact that this must be done with care in order not to undermine the security

measure entirely. Claim 1 of the fourth auxiliary request thus lacks inventive step, too.

Fifth auxiliary request

21. Claim 1 of the fifth auxiliary request specifies two alternative exceptions to the strict security policy, one of them being to allow the storage of executable files "under [...] a trusted system account". It is, however, in the board's view, common place for security reasons to give the security administrator more rights than end-users. In the present case it is obvious in particular to allow the system administrator, typically occupying a "trusted system account", the right to install software and, in doing so, to store executable files "in the data storage". Claim 1 of the fifth auxiliary request thus lacks inventive step, too.

Summary

22. There not being an allowable request, the appeal must be dismissed.

Order

For these reasons it is decided that:

The appeal is dismissed.

The Registrar: The Chairman:

B. Atienza Vivancos W. Sekretaruk

Decision electronically authenticated