Jestr

Journal of Engineering Science and Technology Review 5 (1) (2012) 63-71

JOURNAL OF
Engineering Science and Technology Review

Review Article

www.jestr.org

Overview of Real-Time Antivirus Scanning Engines

L. Radvilavicius*, L. Marozas and A. Cenys

Information Security Laboratory, Dept. of Information System, Faculty of Fundamental Sciences, Vilnius Gediminas Technical University
Sauletekio al. 11, SRL-I-415, LT-10223, Vilnius, Lithuania

Received 16 March 2012; Accepted 10 July 2012

Abstract

Malicious code is one of the biggest problems in the world of networks. There exist various methods and techniques stating that they protect user. For quite some time the most popular protection method against viruses was on-demand scans. Various attempts to implement on-access or real-time scanning mechanisms were either consuming too much valuable system resources such as memory or offering too little protection.
In this article we review a number of open-source and patented methods of real-time antivirus scanning describing their methods of work, advantages and disadvantages. Such kind of research is needed in order to gather in one article and to demonstrate the methods and attempts to successfully implement real-time scanning mechanism and to overview this sphere of application development. In what direction could the next step in developing real-time antivirus scanners be made and what problems are common in such cases

Keywords: Real-time, antivirus, on-access, scanner.

1. Introduction

Viruses, worms, malware and other sorts of malicious code are one of the biggest threats to computer systems since the first virus detected in 1970s. In the current age of the Internet they spread and propagate faster and easier than ever before despite the improvements in antivirus software and their wide choice. Two most important factors from the end-user's point of view are performance and antivirus software ability to quickly and correctly distinct infected files from the healthy ones. Two types of scanner engines are used by antivirus applications –on-demand scanners that are activated only by user in order to scan a part or all of computer and on-access (otherwise called real-time, background guard, resident shield, autoprotect) scanners that monitor data real-time i.e. while data is coming into computer, files are being opened and during similar actions. In latter case when malicious activity is detected, the antivirus system is able to block it before it does any harm to computer system. Another field of modern technology that is prone to malicious activities is mobile technologies. While phone is not only a tool, but has become a part of business life, it usually consists of most sensitive information and thus became a target of malicious software and attackers. Having in mind the power capabilities and memory resources on these devices, antivirus software must find a way to secure people using modern mobile operating systems such as Android.

Most modern antivirus systems offers real-time protection for their users since monitoring and analyzing files while they are being accessed lets protect the user better than on-demand scans. Different antivirus systems have different methods for doing that, but the main disadvantage of real-time monitoring is high system resource consumption. According to performance tests in [1], application launch time with functioning antivirus software might be extended up to almost 5 times and idle memory usage – up to almost 150 times! Thus the creators and designers of antivirus software must find the golden mean between the effectiveness of antivirus scanner and consumption of system resources. It is not possible to analyze and present the methods that commercial antivirus systems are using for they are patented, closed source and hidden from the public eye. That's why in this article there are also several patents are overviewed, described and presented in order to have an idea how the engines work and the ways how commercial and/or closed source antivirus system real-time scanning engines function.

Our goal in this article is to review some of existing open source real-time scanning engines, to analyze the way they are performing and to present their differences. We are also presenting several patents and the principles of how commercial antivirus systems are designed. In the final chapter we are presenting researches that were done in mobile technologies in order to protect mobile users from viruses and malware.

* E-mail address: lukas@fmf.vgtu.lt
ISSN: 1791-2377

2. Open source real-time scanning engines

Viruses are capable of various actions when they are able to penetrate the system. This includes any sort of undesired by end-user activity from simple propagation to sending out contacts, damaging operating system and so on. This is why it is of utter importance to block such activities before they are capable of doing any damage. Many antivirus systems perform scanning or takes action only when post factum – when file is opened, closed or executed. It is inefficient in respect of used resources and it is even worse that they may detect virus after it was already executed or written to stable storage. Y. Miretskiy et al. [2]. This is the reason why the true power of antivirus software lies in real-time scanner engines they are using, it's ability to balance used resources and to perform their actions silently, but efficiently.

2.1 Stackable file system – Avfs and Oyster virus scanning engine

Avfs is stackable file system that is based on open-source ClamAV scan engine. Stackable file systems offer easier development of file systems by offering mechanisms for incremental development as stated in [3] by E. Zadok et al. They are kernel-resident file systems that are based on Virtual File System (VFS) thus they don't fall neither in category of native file systems that directly interact with low level media though are difficult to develop and debug, nor user-level file-systems that usually suffer from poor performance due to the number of context switches serving user requests. E. Zadok et al. [4]. In case of Avfs, it is mounted over existing file-system and becomes a bridge between existing file system and VFS (see fig. 1). Avfs performs virus scanning and state updates during calls of VFS that are passed to corresponding underlying file system via Avfs bridge. From Y. Miretskiy et al[2]. Oyster is a virus-scanning engine, residing in Linux kernel that exports API used for scanning files and buffers of data. Virus scanning is performed during individual page reads and writes.

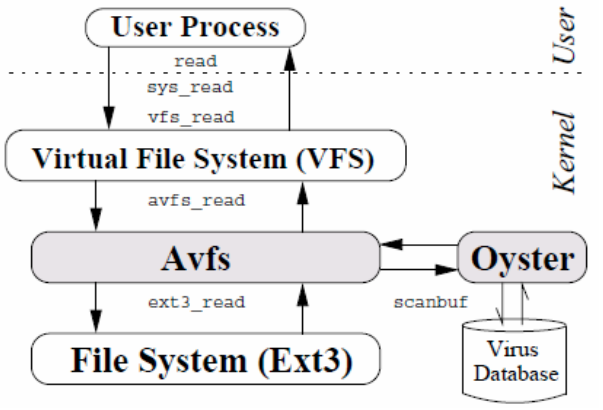

Fig. 1. High level view of Avfs infrastructure [1]

Oyster is an enhancement to ClamAV virus scanner. Firstly it removed scalability problems that were present when dealing with ClamAV scanner by redesigning data structures. Secondly it improved memory usage by modifying trie construction algorithm. And thirdly kernel integration was added. Oyster uses data units native to kernel and does not scan entire files as ClamAV scanner does. Avfs as on-access scanning "addition" to Oyster engine performs partial and non-repetitive scanning. It has two states for performing real time virus protection. One allows access to files through *read* and *write* methods, tracking patterns across page boundaries. It's computed by Oyster engine and maintained by Avfs. Second state, in order to avoid repetitive scanning, is stored as part of file by Avfs. Y. Miretskiy et al [2]. State design divides file into two parts. Oyster, as well as Avfs has separate methods for storing these states. Operations on files are shown in fig. 2. Simplified Avfs operations on files Avfs are as follows:

1. Unknown file opened. No state associated.
2. Page 1 is read, data scanned, state S1, corresponding to the page is computed by Oyster.
3. Reading next page. S1 used for scanning.
4. File closed. Serialized form of state is stored in state file.
5. Another opening of file. State being brought back to memory.
6. The file is scanned completely or sequential read according to previous state.
7. If the file is scanned completely, the latest state is written and the file is marked clean and won't be scanned unless modified.

Two scanning modes exist – full mode that scans for all patterns in Oyster's database and regular states Y. Miretskiy et al [2]. The latter is faster though less accurate. Also two forensic modes exist – immediate and deferred. Immediate mode returns error to process if malicious activity is found (e.g. infected file is being read) thus not letting virus to be written to disc. Deferred mode records malicious activity and defers error notification. Both modes quarantine the files and denies access to them. Any combination of scanning and forensic modes can Avfs be mounted with. Various tests and benchmarks were shown in Y. Miretskiy et al. [2] that shows that this engine is capable of decent performance though still needs optimization for it is still slower than optimized commercial engines. This was the very first implementation of on-access state-oriented anti-virus solution, scanning input files on reads and writes.

2.2 Hash-AV

Hash-AV is a technique that uses bloom filters and hash functions that fit in L2 cache and accelerates the virus scanning for it does not require direct access to main memory. Bloom filters were first introduced in 1970s by Burton Bloom. It is a simple space-efficient randomized data structure for representing a set in order to support membership queries as stated by Andrei Broder et al in [5]. When string X is given, Bloom filter computes k hash functions on it producing k hash values from 1 to m. Then it sets k bites in a m-bit long vector at the addresses corresponding to the k hash values. Same procedure is repeated for all members of the set. In S. Dharmapurikaret al [6]. Data structures used by multi-pattern string matching algorithms cannot be fit in CPU cache and remain in main memory. Main idea behind hash-AV is to exploit the speed gap between access to main memory and L2 cache (in O. Erdogan et al [7]). It uses a filter that fits in cache as a first-pass scan in order to determine if data need to go through the further algorithm.

Hash-AV constructs a bloom filter from the set of plain-text signatures. Bloom filter is a vector of N bits, all set to 0. For each plain-text signature, k number of hash functions applies to first portion of bytes with results of hash functions in range 1 to N. The bits at the positions are then set to 1.

When scanning, Hash-AV moves over the input data stream in a step of one byte at a time. For each byte block, the algorithm applies the first hash function and then compares the bits with the bloom filter. If the bit is 1, then it goes to the next hash function, if not – to the next byte and starts applying hash functions for the next byte block. O. Erdogan et al[7]. When all functions have positive filter matches, Hash-AV pre-constructs a "secondary hash table" with the last hash function, holding a linked list of signatures. It is important how to set up the Hash-AV. Experiments showed that 4 hash functions do good (J. A. L. Fan et al[8]). so it is important to choose these hash functions, the size of bloom filter and the number of bytes hashed in prefix. Selections were made in [7]. Mask, xor+shift, fast hash and sdbm were chosen for hash functions. To select a size of bloom filter is rather difficult for it depends on a couple of factors such as relative ratio of cache size and the size of the filter. Speed test while choosing the size of byte block hashed in prefix is presented in fig. 3.

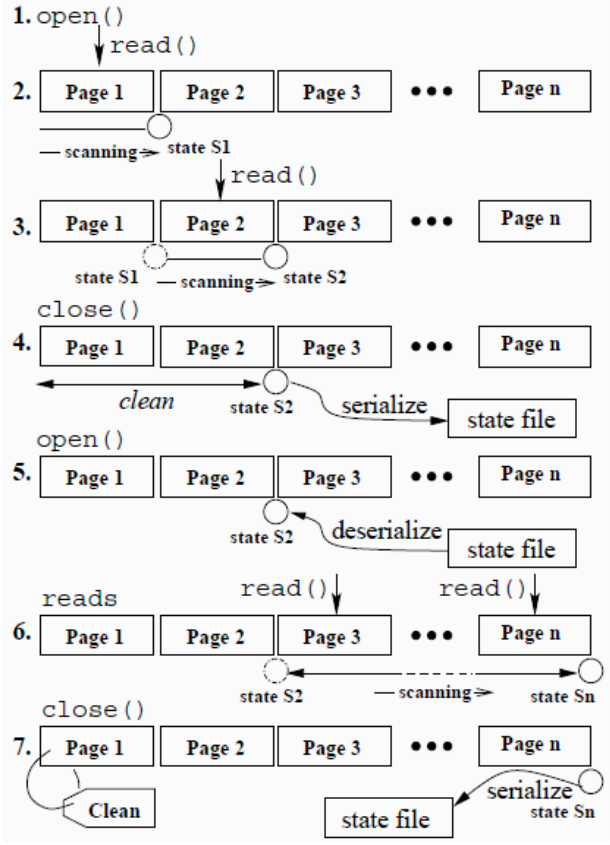

Fig. 2. Operations on files

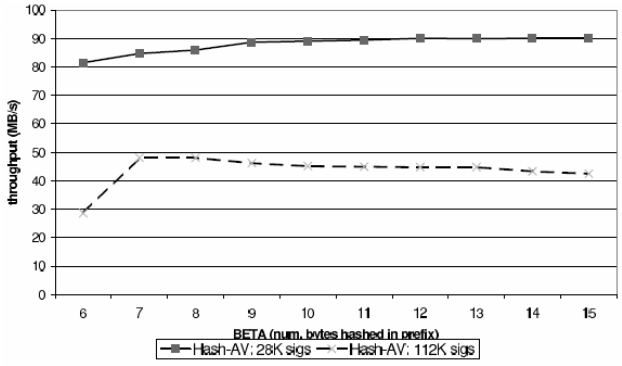

Fig. 3. Performance of Hash-AV for different prefixes [6]

Tests, made in O. Erdogan et al [7] showed that on-access scanner can examine input stream at throughput of over 200Mb/s thus making this technique suitable for network-based on-access scanning. However, when using Hash-AV with ClamAV and the front-end bloom filter fails, it still relies on ClamAV scanner to perform the exact matching. Nen-Fu Huang et al [9].

2.3 Dazuko

Dazuko is an open-source –project. It's an interface for third party applications to control file access (it cannot scan for any sort of viruses itself).J. Ogness [10]. Dazuko works directly with operating system kernel to intercept file-accessing system calls. Application using Dazuko interface first registers and communicates with Dazuko that it is ready to execute file access control. Once the file access occurs, Dazuko notifies the application by sending file name, number of flags etc. Antivirus software in such situation scans the file and allows or denies access to it depending on the scan results. Depending on the decision of software, Dazuko notifies operating system to either continue the process or to return an error. It is important that Dazuko operates in transparent mode thus the application or process receives error codes in case of viruses from operating system. This interface is also capable of working with multiple threads from the same application (J. Ogness [10]). Also it supports cascading that allows different applications to run at the same time as shown in fig. 4.

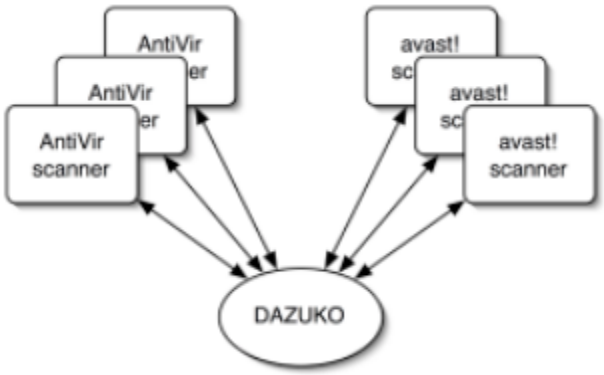

Fig. 4. Two applications, each with three processes, utilizing Dazuko

Dazuko has three layers (see fig. 5). In platform-dependent layer a set of functions according to the platform are implemented. This is the layer that interacts with operating system and since Dazuko is cross-platform, this layer has to be adapted for every supported operating system. Functionality layer is responsible for the decisions. The visible layer is the one accessible to applications. It only provides a front-end for functionality layer to communicate and exchange information with antivirus application.

Dazuko, being still rather young project, is not yet ported to most popular operating systems as Mac OS X and Microsoft Windows. It still needs to improve the security model since it relies on root privileges and thus applications are automatically trusted. The more robust method is in the future works (J. Ogness [10]).

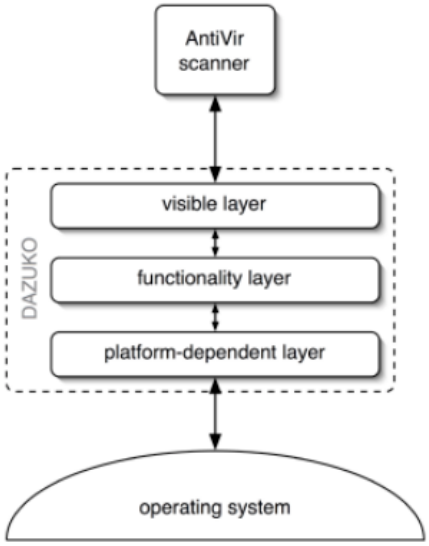

Fig. 5. Layer scheme of Dazuko

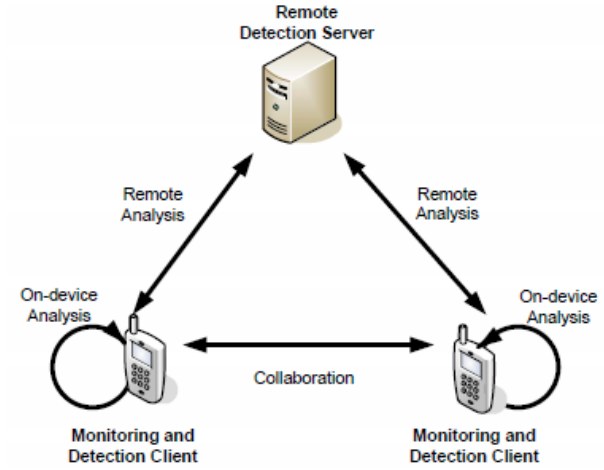

Fig. 6. The proposed system architecture [15]

3. Antivirus scanners for mobile technologies

Nowadays it is important not only to save data that resides on computer systems, but also on mobile devices such as mobile phones etc. These devices usually lack the tools that could guarantee the safety of data. It is also important to have in mind capabilities of these devices since they are not as powerful as computer systems. We think that it is important to overview researches that have been made in this field.

Signature based malware detection method was proposed in D. Venugopal et al [11]. It requires little memory and is suitable for mobile devices though after comparison, the main minus of signature-based detection method was visible – the method was not able to detect zero-day malware.

In H. Kim et al [12] monitoring, detection and analyzing malware-detection software was proposed. The framework is composed of power monitor which collects power samples and builds power consumption history from collected samples and data analyser that generates a power signature from the constructed history. Authors says that malware, especially zero-day is usually hard to detect thus signature-based methods are not the solution. They propose power-aware malware-detection framework that monitors and detects previously unknown energy-depletion threats.

J. Cheng et al [12] developed a system that uses system and log monitoring for malware infection detections. When infections are detected, system alerts the device where monitoring client is installed. The problem that appears is that malware can notice that their activities will be logged.

T. K. Buennemeyer et al in [13] – a way to monitor current changes on smartphones in order to detect anomalies that can be malware or flooding or so is presented. Monitored data is sent to remote server that is able to detect anomalies.

A. Bose et al [15] propose behavioral detection framework. They represent malware behavior based on an observation of applications revealing their malicious intents over time. Two-stage mapping technique, constructing these behavior signatures was proposed.

A. D. Schmidt et al in [16] a monitoring and detection client-server system was proposed for Android based devices. It provides three main functionalities: on-device analysis, collaboration and remote analysis. The principal scheme of system architecture is shown in fig. 6.

The client of monitoring and detection architecture is shown in fig. 7. The Linux application level provides the functionality for monitoring and storing device and operating system information while java application level anomaly detection, collaboration and response actions are realized.

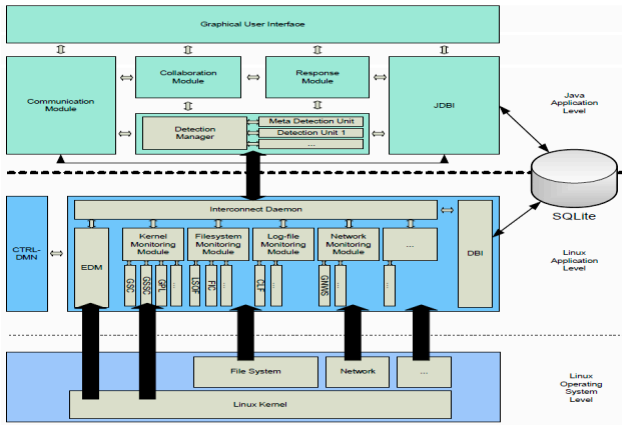

Fig. 7. Client architecture

4. Patented techniques of real-time antivirus scanning

There exist a number of antivirus real-time scanning techniques that are patented and not distributed under GPL as open source. Since it is not quite possible to clearly and correctly describe the methods, used by commercial antivirus systems, in this section we are reviewing some important patents that are presenting innovative ways for real-time monitoring and scanning from malicious code.

In USA patent [17] Kasperski lab patented method and system for antimalware scanning. Invention, registered in 2010, provides the solution for scanning executable files for malware presence. The flow chart of the invention is shown in fig. 8.The invention asserts it reduces the scanning time while balancing quick (but usually less thorough) checks with more exhaustive and thus slower ones. Request for scanning must pass a number of processes and only after that the system is granted with access to it. Large files are treated separately.

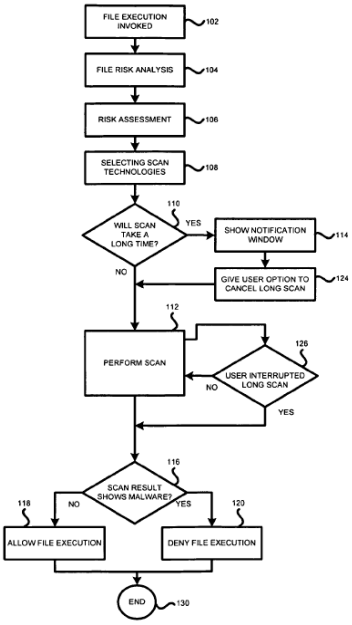

Fig. 8. Scheme of patent [17]

In Microsoft patent [18] identification of malware that is loaded in a memory is published. Software routines that are implemented in this invention, track the state of pages that are loaded in memory. The scheme of the patent is shown in fig. 9.This invention is capable of detecting malware regardless of how it accessed the device (avoided detection using encryption, exploited an application that was already in memory etc.). According to the specifications, this technique offers a possibility for performance improvements of antivirus software. All programs loaded into memory are stated as unsafe and possible threats. Thus the system calls a scan engine to search for infections before they are executed. It is also capable of detecting unknown malware using heuristics for it scans the memory for malicious activity prior to the execution.

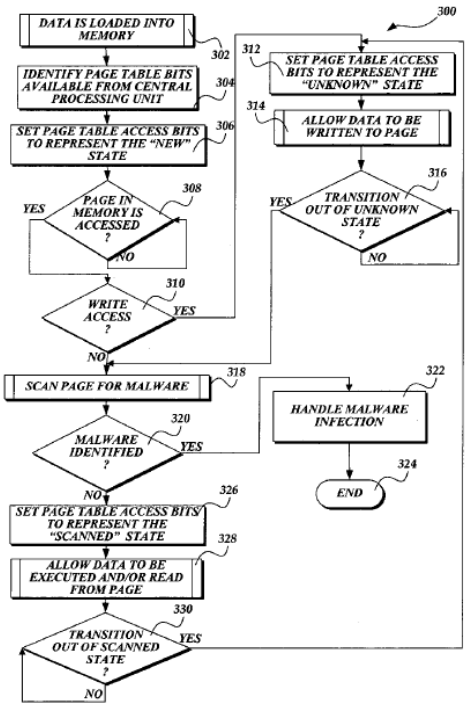

Fig. 9. Scheme of patent [18]

Patent [19] states that the invention reduces file access time during real-time scanning through predictive preemptive scanning. Invention was registered in 2010. The predictive scanning is possible because of file access performance cost mapping in the file system. It can be developed by monitoring time taken to scan the file and so on. At first file access information is collected. The second step is to generate time cost statistics for accessing files. The next step determines the frequency in which a file was accessed. After these steps, file access cost mapping is generated and files may be pre-scanned thus reducing the scan time needed for on-access scanning engines. The full scheme of patent is presented in fig. 10.

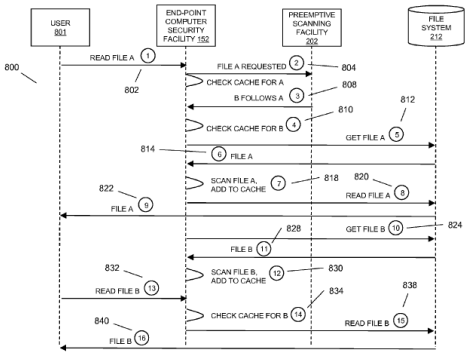

Fig. 10. Scheme of patent [19]

In USA patent [20] technique that is capable of detecting malware in compressed or emulated files is presented. This method interrupts execution of a process, scans process memory for malware at first and then allows or terminates it depending whether malware is found or not. These processes can be associated with the application and loaded from compressed or encrypted file. One file that is not needed to perform decryption, decompression or unpacking may be infected. Flow diagram of this invention is shown in fig. 11.

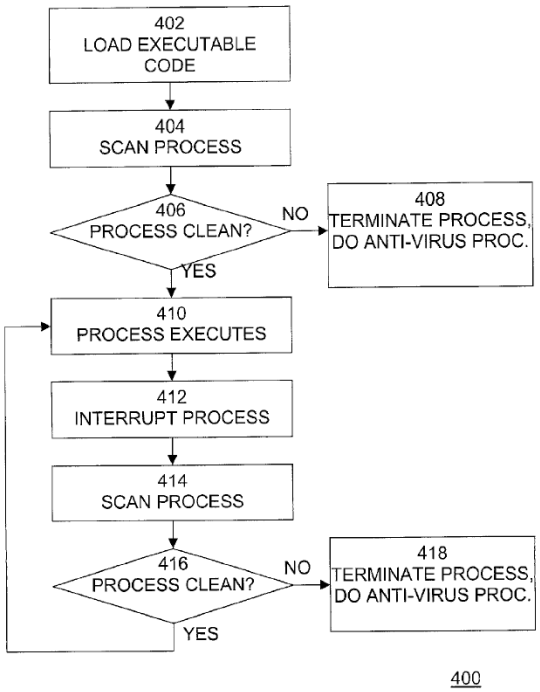

Fig. 11. Flow diagram of patent [21]

In 2005 the USA patent [21] was registered. This malware detection system is based on comparing the checksums of candidate file (possibly infected) with the checksum of healthy file. If they match – the candidate file is thought to be safe. Inventors state that by identifying some of the well-known files as virus-free, especially complex ones, can significantly boost up the performance of antivirus software. The attributes calculated from such files helps to insure that candidate file has not been changed (e.g. operating system files etc.). The principal scheme of this invention is presented in fig. 12.

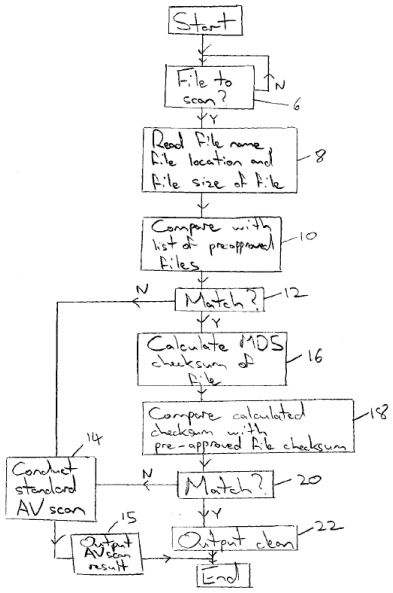

Fig. 12. Scheme of the patent [22]

In USA patent [22] yet another method for boosting antivirus scanning and real-time monitoring performance is presented. Invention states that by using registries, number of files is identified and scanned from malware. Then the registry can be monitored for identifying changes. The performance is boosted because of decrease of scanning time that is needed to scan registry entries in contrary to scanning files on hard disks. Scheme of the patent is presented in fig. 13.

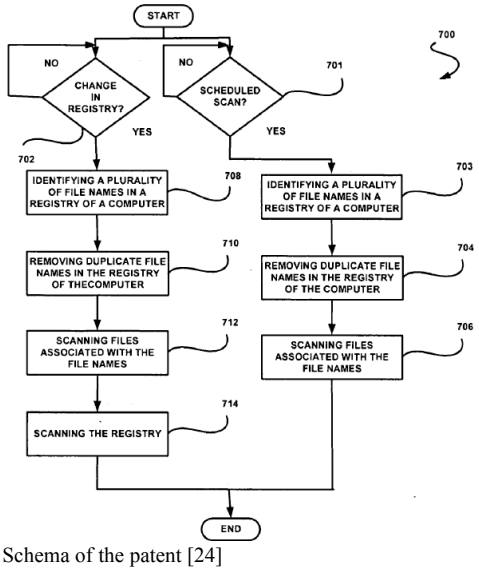

Fig. 13. Schema of the patent [24]

In 2006 USA patent [23] was registered. The main idea of this invention is to identify attempts to change the registry and thus prevent it. Prevention is processed according to a set of rules. They may involve name of a process that requests a change, name of a value in the registry and so on. The principal scheme of this rule-based method is shown in fig. 14.

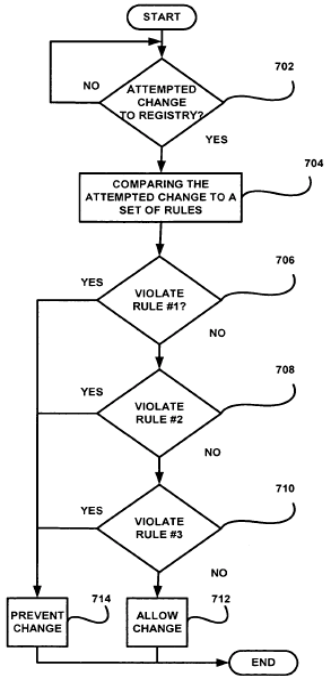

Fig. 14. Scheme of the patent [25]

Yet another interesting invention was patented in USA patent [24]. In this invention, virus scanning capabilities are added to a data transfer device (e.g. controller). In such case, this real-time scanning mechanism is capable of scanning data when it is being written to a file. In case malicious code is detected, antivirus system is invoked that scans the supposedly infected file. Inventors state that this invention greatly reduces the resource consumption of the system. The scheme of patent is presented in fig. 15.

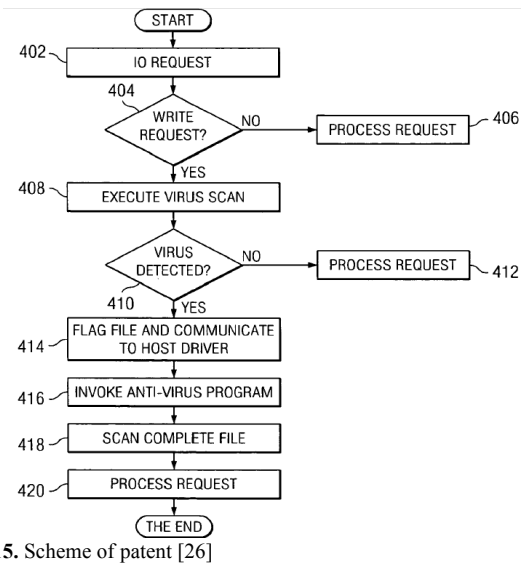

Fig. 15. Scheme of patent [26]

Efficiency of real-time scanning technique was improved and presented in USA patent [25]. This invention is mainly based on identifying processes that access files. If process doesn't have an identifier, virus detection is selected in part on the identification process. When the identifier is assigned, virus detection might be selected in part on the identifier thus accelerating the whole process. The identification and analysis of process every time is avoided. When the process is terminated, the identifiers are cleared. The scheme of the patent is present in fig. 16.

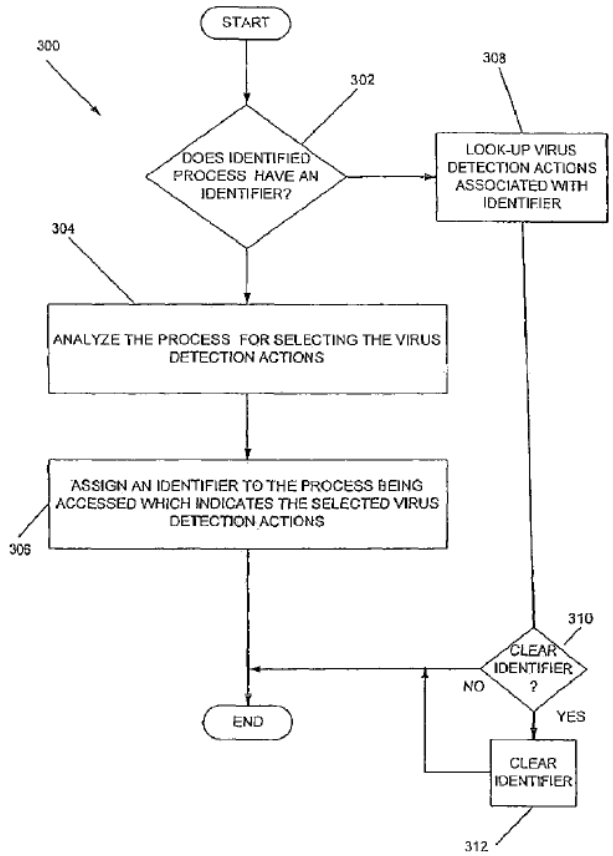

Fig. 16. Scheme of patent [28]

In USA patent [26], registered by Symantec Corporation in 2010, antivirus scan of file in real-time is presented. When the activity, initiated by a first thread, associated with a file, is detected by thread manager, it determines that a scan should be performed. It is initiated by the thread manager as a second thread enabling the first thread to complete actions without delays, but access to the file is blocked while the scan is performed. The scheme of this patent is presented in fig. 17.

In yet another patent by McAfee [27], a real-time scanning mechanism without significant loss of performance is presented. It provides delayed file write operation. This technique intercepts a file access operation of a process to file. Then it waits an interval of time between intercepting and scanning a file for malware. After the period of time, scanning of file is performed. The file write operation that was intercepted and the operations, associated with the file are monitored and are allowed to complete before or during the scan. The time interval may be user-defined. It can be based on a file type. It also can be based on process. The scheme of the patent is presented in fig. 18.

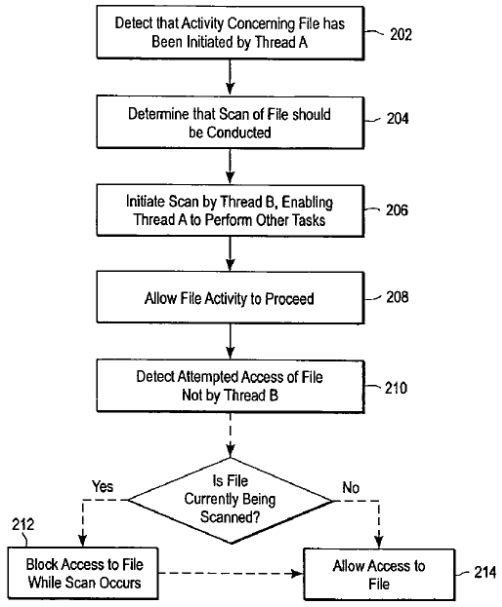

Fig. 17. Scheme of the patent

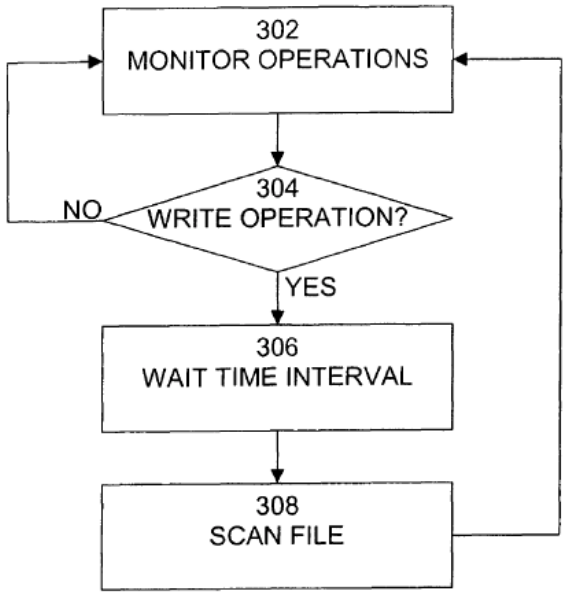

Fig. 18. Scheme of patent [30]

4.1 Patented techniques of distributed real-time antivirus scanning

In 2007 McAfee patented a technique [28] for dividing and distributing scanning tasks if they have the complexity above a specific threshold level. The request to perform an on-access scan is divided into separate files that can be scanned either as different tasks or sent to different computers for scanning. After the scans are completed, results are returned to the computer that initiated the scan and the scan result is formed. There are several techniques for dividing the files offered (e.g. to divide the file into several component computer files like ZIP etc.). This invention also deals with the problems of communication between computers. The scan is interrupted if the part of the file in one of the computers is infected. The principal scheme of this invention is shown in fig. 19.

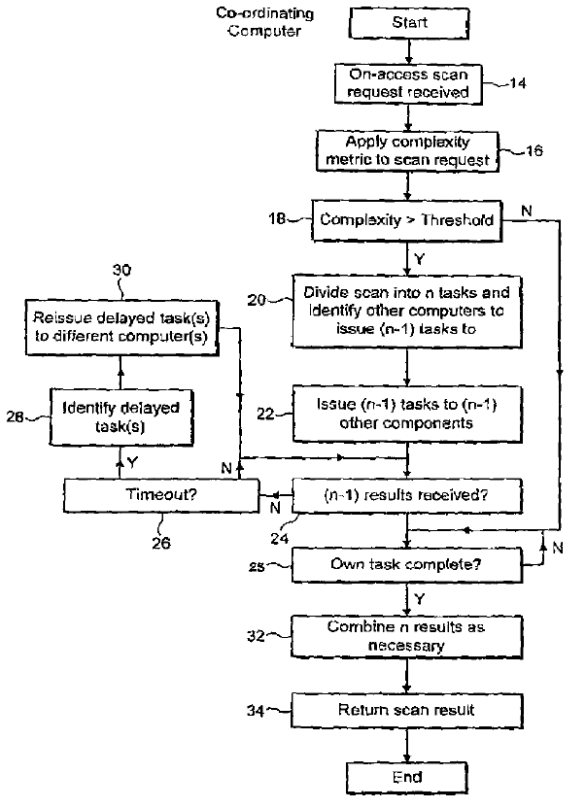

Fig. 19. Scheme of patent [20]

In [29] yet another technique of distributed scanning through several virus checkers is presented. Because of the opposite nature of on-demand and on-access virus scanning, these two techniques are both grouped into chunks and placed on queue. But chunks of on-demand scanning are not placed on queue unless there exist on-access requests. The principal scheme of the method is shown in fig. 20.

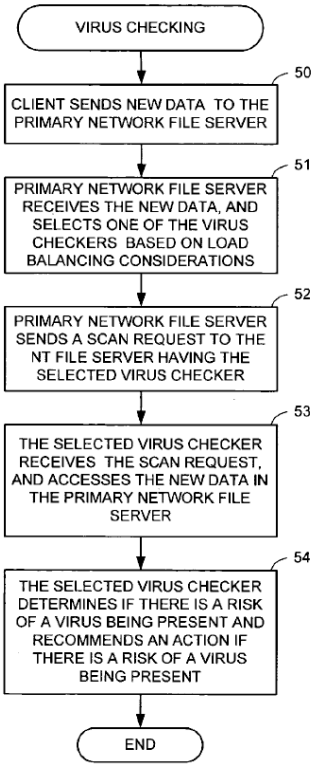

Fig. 20. Principal scheme of the paten [23]

In [30] an on-access scanning technique for scanning archives that may contain malicious code is presented. This invention determines client session characteristics for connection with server. Unshared application, such as antivirus, is a scanner that performs on-access scanning and allows the virus scanner to notify the client of appearing problems. This technique also monitors and determines if the scanning time does not take too long. In case scanning operation takes place in terminal server environment, it is capable to identify client connections to the server for error messages to be presented on the client's terminal and not on server's. The principal scheme of the patent is presented in fig. 21.

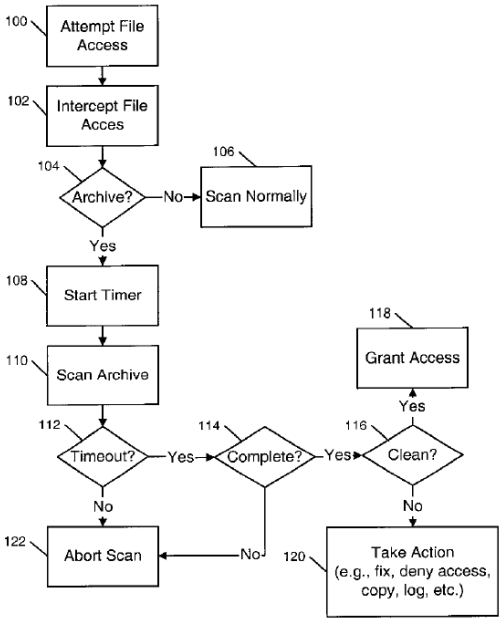

Fig. 21. Scheme of patent [27]

5. Comparative analysis of open source real-time scanning techniques

Most of the antivirus software presented in this article is commercial, closed source and their operating techniques are kept as a secret so it is quite impossible to determine by what methods they are working. Despite this fact, the main idea about their functionality and how they operate can be discovered while looking through white papers and patents that are presented above (Table 1).

6. Conclusions

Since the field of on-access or real-time antivirus scanners are one of the most important factors when stopping malicious software, on-access antivirus scanning engines were reviewed and analyzed.

Positive and negative aspects of methods were analyzed and reviewed

After reviewing real-time monitoring and scanning engines, it is clear that there is still plenty of room for improvement and this field is the perspective one while talking about antivirus software.

Monitoring, malware and anomaly detection engines for mobile technologies were reviewed and their positive and negative aspects described.

Table 1. Comparative analysis of open source real-time scanning techniques

Method	What is it?	What is it based on?	Performance	Advantages	Disantvantages	Scanning performed	Technique used
Avfs	Stackable file system	Clam-AV	Better than Dazuko. No information when comparing it with Hash-AV	Flexible, fast,	Scan engine not quiet optimized, the whole files are scanned,	Reading, writing	Signature-based
Hash-AV	Improved virus scanning technique	Clam-AV	No information when comparing with Avfs. Not comparable with Dazuko	Greatly improved performance, quite flexible	Values must be carefully chosen, still needs improvements	Open, exec, closed, glibc wrapping	Signature-based
Dazuko	Kernel module/interface for antivirus to communicate with operating system	Clam-AV	Slower than Avfs. Not comparable with Hash-AV	Universal, Difficult to use in most popular operating systems	Security issues	Open, exec, close	---

Number of patented, commercial or closed source techniques was presented in order to acknowledge the reader with variety of methods and techniques used in real-time scanning.

References

1. http://www.raymond.cc/
2. Y. Miretskiy, A. Das, C. P. Wright, E. Zadok “Avfs: An On-Access Anti-Virus File System”, In proceedings of the 13th USENIX Security Symposium, 6-6, 2004.
3. E. Zadok, J. Nieh ”FiST: A Language for Stackable File Systems”, In proceedings of the 2000USENIX., 2000.
4. E. Zadok, I. Badulescu “A Stackable File System Interface For Linux”, In Linux Expo Conference Proceedings, 141–151, 1999.
5. A. Broder, M. Mitzenmacher “Network Applications of Bloom Filters: A Survey”, Internet Mathematics, 1-4, 2003.
6. S. Dharmapurikar, P. Krishnamurthy, T. Sproull, J. Lockwood “Deep Packet Inspection using Parallel Bloom Filters”, In 11th Symposium on High Performance Interconnects, 2003.
7. O. Erdogan, P. Cao “Hash-AV: Fast Virus Signature Scanning by Cache-Resident Filters”, In Proceedings of Globecom'05, 2005.
8. J. A. L. Fan, P. Cao and A. Broder. “Summary cache: A scalable wide area web cache sharing protocol”, In Proceedings of the 1998 ACM SIGCOMM Conference, 1998.
9. N.F. Huang, W.Y. Tsai “SHOCK: A Worst-Case Ensured Sub-linear Time Pattern Matching Algorithm for Inline Anti-Virus Scanning”, Communications (ICC), 2010 IEEE International Conference, 1-5, 2010.
10. J. Ogness “Dazuko: an open solution to facilitate on-access scanning”, Virus Bulletin, 2003.
11. D. Venugopal, G. Hu, “Efficient signature based malware detection on mobile devices”, Mobile Information Systems, 4, 33–49, (2008).
12. [12] H. Kim, J. Smith, K.G. Shin, “Detecting energy-greedy anomalies and mobile malware variants: In Proceeding of the 6th international conference on Mobile systems, applications, and services, 239–252, 2008.
13. J. Cheng, S.H.Y. Wong, H. Yang, S. Lu, “Smartsiren: virus detection and alert for smartphones” In International Conference on Mobile Systems, Applications, and Services, 258–271, 2007.
14. T.K. Buennemeyer, T.M. Nelson, L.M. Clagett, J.P. Dunning, R.C. Marchany, J.G. Tront, “Mobile device profiling and intrusion detection using smart batteries” In HICSS ’08: Proceedings of the Proceedings of the 41st Annual Hawaii International Conference on System Sciences, 296, 2008.
15. A. Bose, X. Hu, K.G. Shin, T. Park, “Behavioral detection of malware on mobile handsets” In Proceeding of the 6th international conference on Mobile systems, applications, and services, 225–238, 2008.
16. A.D. Schmidt, R. Bye, H.G. Schmidt, K.A. Yüksel, O. Kiraz, J. Clausen, K. Raddatz, A. Camtepe, S. Albayrak „Monitoring Android for Collaborative Anomaly Detection: A First Architectural Draft“,Technische Universita□t Berlin DAI-Labor, http://www.dai-labor.de, 2008.
17. Method and system for antimalware scanning with variable scan settings, United States Patent 7725941.
18. On-access scan of memory for malware, United States Patent, 0200863.
19. Method and system for preemptive scanning of computer files, United States Patent, 0242109.
20. Method and system for detecting computer malware by scan of process memory after process initialization, United States Patent, 2003/0115479.
21. Pre-approval of computer files during a malware detection, United States Patent, 2005/0021994.
22. System, method and computer program product for accelerating malware/spyware scanning, United States Patent, 2006/0075502.
23. System, method and computer program product for preventing spyware/malware from installing registry, United States Patent, 2006/0041942.
24. Method and system for offloading real-time virus scanning during data transfer to storage peripherals, United States Patent, 2006/0156405 .
25. System, method and computer program product for selecting virus detection actions based on a process by which files are being accessed, United States Patent, 6931540.
26. Semi-synchronous scanning of modified files in real time, United States Patent, 7681237.
27. Method and system for delayed write scanning for detecting computer malwares, United States Patent, 7757361.
28. On-access malware scanning, United States Patent, 7243373.
29. On-access and on-demand distributed virus scanning, United States Patent, 2005/0149749.
30. Obtaining user responses in a virtual execution environment, United States Patent, 6594686.