Vulnerability dependencies in antivirus software

Kreetta Askola, Rauli Puuperä, Pekka Pietikäinen,
Juhani Eronen, Marko Laakso, Kimmo Halunen, Juha Röning
Oulu University Secure Programming Group
Department of Electrical and Information Engineering
P.O. Box 4500
90014 University of Oulu
Email: ouspg@ee.oulu.
Finnish Communications Regulatory Authority FICORA
Itämerenkatu 3a
00181 Helsinki
Email: juhani.eronen@ cora.

Abstract

In this paper we present an application of the MATINE method for investigating dependencies in antivirus (AV) software and some vulnerabilities arising from these dependencies. Previously, this method has been effectively used to find vulnerabilities in network protocols. Because AV software is as vulnerable as any other software and has a great security impact, we decided to use this method to find vulnerabilities in AV software. These findings may have implications to critical infrastructure, for the use of AV is often considered obligatory. The results were obtained by gathering semantic data on AV vulnerabilities, analyisis of the data and content analysis of media follow-up. The results indicate, that different aspects of AV software should be observed in the context of critical infrastructure planning and management.

Key words: Vulnerability dependencies, dependency tracking, antivirus vulnerabilities

1 Introduction

Vulnerabilities are abundant in modern software intensive systems. Bugs and security flaws can also be found in the very software that is supposed to keep one safe from malicious programs (malware). The use of antivirus (AV) software is widely adopted procedure also among critical infrastructure systems [8].

However, protecting oneself from malware is not that simple. First, although AV software is considered to increase security, it is made by the same programming processes, that make insecure programs. In general, any software is breakable [2]. Secondly, AV software population is quite homogeneous, which in itself is a warning sign, as it enables the spreading of malware [1]. The market is dominated by a few leading vendors and using more than one AV program at a time is usually impossible [9]. Homogeneity facilitates the design process of malware, for it is fast to test the malware in all of the most common AV software [12]. Thirdly, AV software require high access rights in order to perform systems monitoring, which makes them attractive attack vectors for systems compromise.

The concept of vulnerability is complex and multiform, for it includes challenges related to permanent existence as well as classification and managing of vulnerabilities. Also the current status of AV software use is a complex phenomenon. The use of AV software does not automatically increase security, but may be a source of unnecessary risk, especially for critical information infrastructure. For example, the main component of an AV software is the scanning engine, which is responsible for identifying malicious files using signature databases. Although some AV software allow use of different engines to enhance protection, many AV software share the same integral scanning engine. [14, 15]

All software contains bugs due to various factors, such as inherent difficulty in translating the requirements to code, complexity of the requirements or the underlying system, immature programming practices and methods [7, 2]. Bugs with security implications are called vulnerabilities. Perpetual vulnerabilities have forced the development of conceptual methods and tools to manage them. Formal and machine-processable taxonomies foster automated analysis and tracking of vulnerabilities. In time, testing would re-

duce the likelihood of the occurrence of the bug type and gradually make it relatively infrequent [4]. It is unclear, whether software is actually improving with respect to these problems, or if they are not inspected as frequently.

Apart from technical vulnerability, there are issues related to vulnerability disclosure and reliability of AV software. AV software vulnerabilities are not in general reported by the media, even though the number of AV vulnerabilities has expanded rapidly in recent years [13]. Although the overall vulnerability numbers seem to have decreased, the future progression of AV vulnerabilities is unpredictable. As it is not easy for the users to test AV products prior to purchases, they are forced to trust the vendors' promises of reliability, for independent assessments of reliability of AV software are rare.

Despite the given problems, AV software is at present considered as a basic element of safe computer use. For example, FICORA[1] recommends that an AV software should be installed to computer systems in order to protect them from malware. HIPAA [8] and Sarbanes-Oxley Act, (SOX) [11] have extended these security requirements to laws. The same conception of security produced by AV software is distributed by security policies, user education and media. There is considerable lack of controversial opinions in all of these areas.

The current security paradigm is the main reason for problems in the context of AV software use. Although AV software increases security for an everyday-user, the necessity of using AV software should be reconsidered in critical infrastructure systems. In many cases, the use of AV software may expose the system to unnecessary vulnerabilities and cause needless dependencies. Many critical systems do not handle the kind of information that AV software is meant to protect.

2 Approach

2.1 Dependency tracking and critical infrastructure in antivirus vulnerability context

In this paper, dependency is de ned as a linkage between entities or common metadata. Dependencies are discovered by forming descriptive metadata and links from given information and then analysing common features and differences of this semantic data. In the case of antivirus vulnerabilities, bene ts from discovering dependencies are multiple. In critical infrastructure, dependencies can be identi ed on multiple levels including technology, functions, people, processes and location.

[1] Finnish Communications Regulatory Authority, http://www. cora. /en/index.html

Dependency tracking has been used in the context of critical infrastructure before. For example, Crisis and Risk Network, (CRN)[2] has published The CRN International CIIP Handbook, which presents national policy approaches to critical information infrastructure protection and the methods and models used to assess the vulnerability and security of these structures. [10]

The concept of meta levels (see Table 1 on page 3) is applicable to any context with inherent dependencies. Meta level is an attribute of a vulnerability, which describes its level of abstraction as well as its scope. Information on the structure of different systems and their relations highlights elements, which are highly connected or common between multiple systems. Vulnerabilities in these elements are typically of a higher meta level as they can result in epidemic failures due to their wide implementation base, or cascading effects due to the failure of a high number of dependent elements. [5]

Meta level zero describes the case where a vulnerability only affects a single implementation (a software version). Meta level one vulnerabilities affect a whole class of systems (all software that implements interface x). Meta level two vulnerabilities affect a super-system consisting of multiple classes of systems (all software having any interface that includes subsystem x). Meta level three affects an element that is used for widely disparate purposes, perhaps by a great number of systems (all systems that use a certain notation, encoding, or other function). [5]

In this study the attention is focused on the le formats that AV software handle. File formats constitute a common public interface to different AV programs, constituting a hothouse of overt and covert dependencies. However, noticing dependencies in this area may be dif cult or even impossible, because same le format can cause same problems in different software and some le formats may include other le formats. Especially in the latter case, the underlying reason may lead to different algorithms in different parsing implementations of le types. In addition, all AV software do not support all formats, for example, the support of archive le formats varies considerably between different software.

2.2 The MATINE model

The research method is based on earlier OUSPG[3] project, PROTOS-MATINE[4] which focused on protocol dependencies and produced the PROTOS-MATINE model [5] (see Figure 1) and the semantic tool Graphingwiki [6],

[2] http://www.crn.ethz.ch/

[3] Oulu University Secure Programming Group, http://www.ee.oulu. /research/ouspg

[4] PROTOS - Security Testing of Protocol Implementations, http://www.ee.oulu. /research/ouspg/protos/index.html

Table 1. Vulnerability Metalevels

Meta level 3	Single scheme in multiple protocols / protocol families
Meta level 2	Single protocol embedded in multiple protocol families
Meta level 1	Single protocol, multiple implementations by multiple vendors
Traditional approach	Single vendor, single implementation, single vulnerability

which are now put into use in the context of AV vulnerabilities.

The model presents an iterative method for rapidly gaining insight on a field of study. The model uses several sources of data, such as specifications, literature, media and experts. All of the gathered information works towards a common goal - understanding a technological subject on multiple levels: its contents and structure, its history as well as projected future, its fields of use and use cases, and its environment and relations to other subjects. With this kind of knowledge, the weight of the subject can be accurately determined in a desired context, such as a system, a network, a corporation or a sector of the critical infrastructure. The MATINE model has been applied in depicting effects of ASN.1 vulnerabilities with heavy emphasis on systems used in critical infrastructures [7, 5].

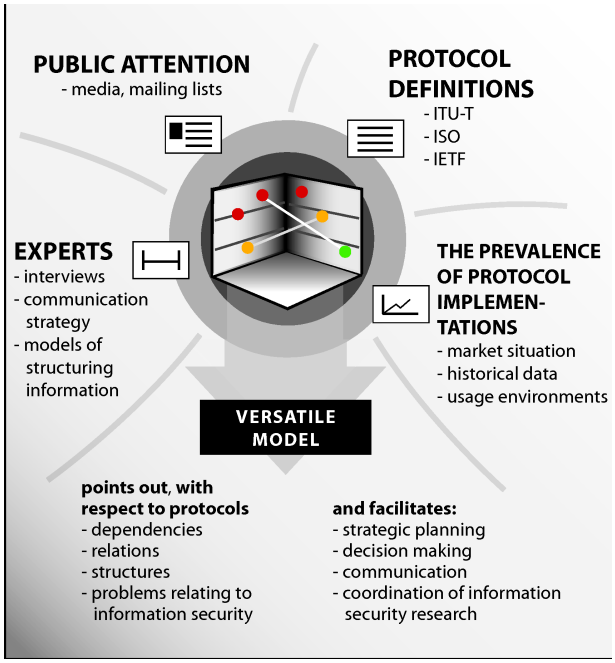

Figure 1. Model for analysing protocol dependencies

In the context of AV software, vulnerability databases and media represent the main data sources of the MATINE model. Media tracking and review of the market situation lay out the priorities of later data gathering and the relative importance of different AV software. Reviews of specifications and expert interviews are considered out of scope for this paper.

The semantic information on AV vulnerabilities, for example impact type and file format, was gathered from National Vulnerability Database (NVD)[5], for it is government-run and thus official and it's descriptors of vulnerabilities are classified and presented in specified, CVE standard form [4]. Additional information was gathered as a media follow-up, which was focused to national level. The media follow-up consisted of regular observation of Digitoday Finland[6], commercial news database focusing on IT sector, throughout the year 2006. News considering AV issues were classified and analysed with content analysis. The focus of media follow-up was on how the AV software and vendors are presented in the media.

3 Results and analysis

This section contains the data in numbers and shares and presents the picture gathered from the media during our research. We collected AV vulnerability data from 1998 to 2008. The number for the year 2008 is just the first quarter of 2008. The total number of vulnerabilities was 276 and the main body of the data was from years 2004-2007. The results are gathered in Figures 2 and 3 and Table 2. In Figure 2 the first number is the number of vulnerabilities associated with that file format and the second number is the procentual share. The number of AV vulnerabilities has expanded rapidly through these years (see Figure 3). The year 2006 was exceptional, as the number of vulnerabilities was lower than the previous year. However, in 2007 there was again an increase and it seems that any future predictions on the number of vulnerabilities would be mere speculation as there is no clear trend.

From our data we noted, that file formats are associated with most of the vulnerabilities (see Figure 2). The most frequent file formats were RAR, CAB and ZIP and altogether archive file formats were present in 70% of all vulnerabilities with file format association. This prompted research in PROTOS Genome -project, where AV software was tested against malformed archive files. The results of this research are reported in [16]. Our analysis suggests,

[5] http://nvd.nist.gov
[6] http://www.digitoday.

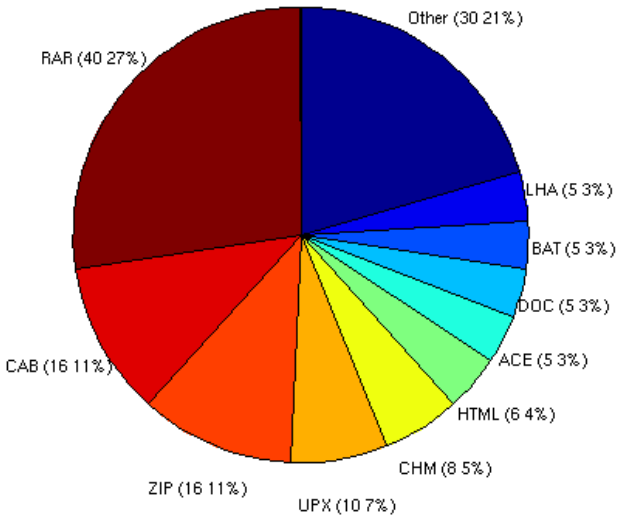

Figure 2. File formats associated with vulnerabilities

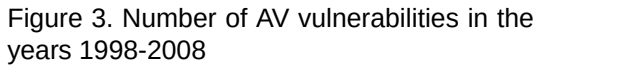

Figure 3. Number of AV vulnerabilities in the years 1998-2008

that biggest factors in the AV vulnerability peak of the year 2005 are different archive file formats, mainly RAR and ZIP vulnerabilities and the results of PROTOS Genome archive test set will affect the number of vulnerabilities in the year 2008.

The most common error type in AV software is design error (see Table 2). Errors in design are hard to avoid, but most of the other types of errors, which account for almost 70% of vulnerabilities, could be avoided by using thorough testing. For example, other most common errors in AV software include buffer overflow, input validation and exceptional condition handling errors. With extensive software testing, the amount of vulnerabilities associated with these errors could be avoided.

The media analysis resource consisted of 92 news items. The results can be seen in Table 3.

In general, AV software is presented in the news in very positive light as continuously developing industry, which provides better solutions and increased security. This is only half of the truth, and the discussion of more negative issues is neglected. For example, in the year 2006 there were 50 antivirus vulnerabilities listed in NVD database, but only 4 of them were reported in the news. The subcontracts are much more visible part of the news. All subcontract news considered new contracts between AV software vendors and various companies such as banks or operators. Apparently the biggest vendors dominate also the news media. However, the vendor shares in news are not representative mainly due to very limited sources included in media follow-up.

From the media analysis the following observations can be made: The AV software is presented very positively, while at the same time vulnerabilities, even the critical ones, are seldom reported. The examined news did not discuss any events on the circulation of code e.g. in the terms of vendor fusions or sharing the engines. Contradiction cannot be found, unless vendor's disputes over vulnerability of operating systems or mobile devices is counted in. The results are promising and support earlier assumptions, but more research is needed for assurance and generalisation.

4 Conclusions

The main goal for this paper is to examine AV software vulnerabilities and the risks they bring to critical information infrastructure systems. The MATINE model was used as a method for disentangling the untrodden field of AV vulnerabilities in a rapid, iteratively expanding fashion. Among the various data sources utilised by the model, public vulnerability data and media sources were tapped by this project. This paper presents the results of our research, which focused on AV software vulnerabilities and dependencies between these vulnerabilities.

One target in the study were file formats and it seems that archive file formats have been the main reason for the fast rise of AV vulnerabilities until the year 2006. Our findings also prompted research in PROTOS Genome -project and the results there show, that archive file formats are still a big issue in AV software. However, the future is unpredictable and it is hard to tell what kind of improvements, if