

Antivirus Software Testing for the New Millenium

Abstract:

The nature of technology is changing rapidly; likewise, the nature of viral threats to the data dependent upon the technology is evolving. Thus, the technologies we rely upon to provide protection from these threats must adapt. In the last twelve months, several anti-virus software vendors have announced exciting new technologies which claim to provide "faster, better, cheaper" response to computer virus incidents within organizations. However, there is currently little guidance regarding the best way to evaluate the efficacy of such claims. Faster than what? Better than what? Less costly compared to what? Clearly, there can only be one technology which is "faster, better, most cost efficient" than all of the others, yet if the advertising claims are to be believed, all products are not merely created equal, they are all created superlative!

In this paper, the requirements for these next generation anti-virus systems will be examined. There will be a discussion of reviewing strategies that can help to determine to what extent those requirements have been met. To this end, the problem will be approached from a functional perspective, not gearing the test design to particular implementations. In this way, an array of tests will be created which are not vendor or product specific, but which can and should be employed industry-wide.

Authors: Sarah Gordon (sgordon@format.com)
 IBM Thomas J. Watson Research Center, U.S.A,

Fraser Howard (fph@format.com)
Virus Bulletin, U.K.

Point of Contact: Sarah Gordon
sgordon@format.com
sgordon@dockmaster.ncsc.mil

Keywords: computer virus, anti-virus product testing, anti-virus product certification, testing methodology, testing criteria, functional requirements.

Antivirus Software Testing for the Year 2000 and Beyond

Sarah Gordon (sgordon@format.com)

Fraser Howard (fph@format.com)

Introduction

In the last twelve months, several anti-virus software vendors have announced exciting new technologies which claim to provide “faster, better, cheaper” response to computer virus incidents within organizations [Anyware, 2000; NAI, 2000; PC-Cillin, 2000; Symantec, 2000a; Symantec, 2000b; Thunderbyte, 2000; Trend, 2000].

However, there is currently little guidance regarding the best way to evaluate the efficacy of such claims. Faster than what? Better than what? Less costly compared with what? Clearly, there can only be one technology which is “faster, better, most cost efficient” than all of the others, yet if the advertising claims are to be believed, all products are not merely created equal, they are all created superlative!

In this paper, the requirements for these next generation anti-virus systems will be examined. There will be a discussion of reviewing strategies that can help to determine to what extent those requirements have been met. To this end, the problem will be approached from a functional perspective, not gearing the test design to particular implementations. In this way, an array of tests will be created which are not vendor or product specific, but which can and should be employed industry-wide.

The State of the Nation: Anti-virus testing in the 90's

Antivirus product testing has improved greatly since the simple zoo scanning offered in the first published reviews. Many, if not most, of the technical and administrative problems documented in [Gordon, 1993; Laine, 1993; Tanner, 1993; Gordon, 1995; Gordon & Ford, 1995; Gordon & Ford, 1996; Gordon, 1997] have been resolved. Today's tests provide a solid, albeit not perfect, measure of product capabilities.

As tests have become more complex, several bodies have emerged as leaders and innovators in this area. Some of the more widely-accepted tests¹ within the industry are outlined briefly below:

(i) *ICSA Certification*

The *International Computer Security Association (ICSA)* has been performing tests of antivirus software since 1992; many popular products are submitted to their various for-fee certification schemes (ICSA 2000). On-access and on-demand scanning are part of their ever-expanding certification criteria; criteria for virus removal were added in July 1999. Primary detection tests are broken into two main sections: In the Wild virus detection, and zoo virus detection. The zoo collection, maintained by ICSA staff is large and fairly complete; products must detect at least 90% of these viruses. Tests on In the Wild viruses now use samples that have been replicated from The WildList Organization's WildCore sample set. These viruses have been confirmed as being an active threat in the user population. To be certified by ICSA, products must detect 100% of these viruses, using the version of The WildList that was released one month prior to the test date. Additionally, a "Common Infectors" criteria ensures that any viruses ICSA feels are important are dealt with in a way ICSA considers appropriate. False alarm testing was added to their testing processes in 1999; Gateway Product criteria were established in July 1998; MS Exchange and Lotus Notes Criteria are being drafted at this time [ICSA, 1999]

(ii) *Westcoast Labs Checkmark*

Westcoast Publishing established itself as a world leader in the testing and certification of antivirus software products in the mid-1990s with its introduction of the *Westcoast Labs Checkmark*. Test criteria depend upon the level of certification applied for. Level One measures the ability of the tested product to detect all of the viruses In the Wild, using samples based upon the edition of The WildList not less than two months prior to the product release date. At Level Two, products must also disinfect these viruses. In addition, the Level Two tests use the version of The WildList that was published one month prior to the product release date. Both tests are carried out using viruses replicated by West Coast, thus measuring the ability of products to detect viruses which constitute the real threat. Many popular products are submitted to this for-fee testing scheme; certified products are announced on a regular basis (Checkmark 2000).

(iii) *University of Hamburg VTC Malware Tests*

Overseen by security and antivirus expert Dr. Klaus Brunnstein, students from the *Virus Test Center (VTC)* at the University of Hamburg have been designing and performing tests of antivirus software since 1994. The results of these projects are made freely available to the general public. These tests have grown from simple

¹ Not all products qualify for testing under all schemata.

tests of boot² and file virus detection in 1994 to the current comprehensive virus (and malware) tests. In addition to their extensive zoo collection, in early 1999, the VTC began using samples replicated from *The WildList Organization's In the Wild* collection in their tests for *In the Wild* viruses, thus assuring (with the exception of boot sector tests) an accurate representation of a product's ability to meet the real threat from these *In the Wild* viruses.

Testing documentation states that users cannot distinguish whether such malevolent software is “just viral” or otherwise dangerous (VTC, 2000a). Thus, the detection of more general forms of malicious software has become a major part of the tests – a decision based upon VTC’s perception of user requirements. These malware tests were initiated in 1998, quickly followed by false-positive testing. While the tests are free, some products are excluded due to various conflicts cited by Professor Brunnstein (VTC, 2000b).

(iv) *University of Magdeberg*

Andreas Marx and his antivirus testing projects for the Anti-virus Test Center at the Otto-von-Guericke University of Magdeberg, done in cooperation with GEGA Software and Medienservice, are relative newcomers to the antivirus testing scene. The tests, sponsored by antivirus companies, provide magazines such as CHIP, FreeX, Network World, PC Shopping and PC-Welt with results; results from these tests have been included in their published reviews. There are seven people involved in the testing process – some students, and some working for the University. According to Marx, most of the test criteria have been chosen by network administrators, users, magazines, AV companies and the University. These criteria include detection of *In the Wild* viruses (Products using the most current WildList), and disinfection (of non-boot sector viruses only). Additionally, non-viral malware tests are carried out as well. Results are made available in both English and German. Participating vendors pay approximately \$300.00 USD per product for testing, all of which is funneled back into the testing project.

(v) *Virus Bulletin*

Virus Bulletin (VB) has been testing anti-virus products since the publication started in 1989. Products for the various platforms are reviewed regularly in the VB Comparative Reviews, which test the products against a zoo collection (standard DOS and Windows file infectors, macro viruses and polymorphic viruses) as well as a recent *In the Wild* set. For each comparative, the Virus Bulletin *In the Wild* set is based upon a version of *The WildList* announced approximately two weeks prior to the product submission deadline; samples replicated from *The WildList Organization's* reference collection are used. In January 1998, the VB100% award scheme was implemented. This award is given to products that detect all of the *In*

² VTC does not use real viruses in boot sector virus testing; they use image files. Their rationale is that it is too time consuming to replicate real boot sector viruses. Their boot sector virus tests are unreliable measures of a product's ability to detect real boot sector viruses. All other testers mentioned do use real viruses in boot sector virus testing.

the Wild file and boot viruses during on-demand scanning. The scheme has grown since then, and now demands complete *In the Wild* detection during both on-demand and on-access scanning. Aside from the detection rate tests, the *VB* comparative reviews also perform tests on scanning speed, on-access scanner overhead and false positive rate. In fact, the “no false positives” criterion is to be introduced into the *VB100%* award scheme for reviews published from June 2000 onwards.

Why this isn't enough from a User Perspective

Given that anti-virus tests have improved dramatically over the last several years as the expertise of the reviewing community has increased, the need for more comprehensive tests may seem unclear. In this section, reasons why tests must move to the next level will be examined.

The much-needed introduction and subsequent development of *The WildList* as a testing criteria provided a reality check to the antivirus industry. With this criterion, users now have a minimum baseline of what any competent (appropriately updated) anti-virus product *should* detect. This criterion is the cornerstone of meaningful antivirus software testing. Indeed, some testers have moved to a one-month WildList, citing the need to show users the ability of products to respond quickly to the ever-changing threat. However, the increased threat from fast-spreading viruses such as Melissa and LoveBug, underlines the need for yet another, more complex shift in focus within testing environments.

The testing industry has also moved forward with various tests related to disinfection, and on-access performance of scanners. As mentioned above, the *VB100% Certification* offered by *Virus Bulletin* recently added on-access tests to their arsenal of stringent antivirus software metrics; both *ICSA* and *WestCoast Labs* have recently implemented disinfection tests. While tests are far from complete, they mirror development of the early *In the Wild* testing.

Clearly, tests are continuing to advance as the industry matures. However, it is not enough to merely expand. As products become increasingly complex, more complex tests are required. This rapidly becomes cost prohibitive; thus, it is important to expand the testing methodology in the areas that are most important to user protection, using metrics that are meaningful both to the users and developers. We propose a new, functionality/requirements based approach, which fulfils the above requirements, and provides excellent return on investment for test costs.

Consider a typical anti-virus product. In essence, most of the protection provided by the product is static: that is, the philosophy behind the product is to detect and (sometimes) remove viruses that are *already known* to the creators of the anti-virus product. However, as was so clearly demonstrated by the explosion of Melissa infections, such an approach is not without risk: as computers become increasingly interconnected, the potential for viruses which spread faster than detection and removal solutions for them can be disseminated is great.

To this end, many anti-virus vendors have added “unknown” virus detection to their products. There are many different subclasses of such generic virus detection, each with their strengths and weaknesses. However, certain facets are universally true:

- Known virus detection is more reliable when dealing with previously known viruses. Thus, generic techniques are unlikely to completely replace traditional signature-based scanners.
- The false positive ratio must be vanishingly small. That is, generic techniques must not mark uninfected, “clean” files as infected. Such mislabeling of files (false positives) can create as much work for a user as a real infection, as the sample must be captured and sent to the anti-virus vendor. The vendor must then either explicitly exclude that file from detection, or retune the entire detection system so that that (and similar) files are not incorrectly singled out for action.

Thus, generic techniques are most effective as a complement to, not replacement of, traditional signature-based techniques.

The next step in the evolution of anti-virus products was a coupling between generic and specific techniques. That is, when a new virus is discovered “in the wild”, the virus is identified and captured generically, and known virus detection is added automatically to provide a “herd immunity” for that virus worldwide. Such a product is therefore neither generic or specific, but *hybrid*, allowing for not just the implementation of both techniques, but for the *integration* of generic and specific detection strategies.

According to several simulations carried out by (Kephart & White, 1993), such a technique would dramatically decrease the opportunity for a virus infection to reach epidemic proportions, as innate immunity to infection would be granted to other machines anywhere in the world as soon as the virus was generically detected on one machine. We have already seen a subtle shift toward this approach, with some vendors offering daily signature file updates via the Internet. However, as computers can exchange data extremely quickly, this process needs to become much more frequent and automated.

Thus, the next generation of anti-virus product must be able to deal with both the known and previously unknown virus arriving at a particular host computer. The virus must be automatically detected in some way, and, when possible and desirable, cured. Furthermore, any other computers which encounter the same virus must be able to immediately and exactly identify the virus - that is, once a single machine has encountered an “unknown” virus, that same virus must be made “known” to other machines.

The last prerequisite is that all of the above must happen quickly; now, how fast is fast enough?

The Melissa incident, which undoubtedly marked a turning point in the anti-virus industry, is a good example of an event in which it is critical to provide a solution quickly. Initially distributed via a posting to the Usenet group ALT.SEX, Melissa

spread perhaps faster than any virus before it. The sheer volume of email that Melissa generated (thanks to its propagating payload which uses MAPI calls to email the infected document to the first 50 entries in the Outlook address book) resulted in the shutdown of countless mail servers. Reports of the generation of between four hundred thousand and half a million email messages within three hours were received [Whalley, 1999]. Clearly, even daily updates of signature files were insufficient to prevent the spread of this virus. Furthermore, Melissa is not the only piece of malware which utilizes email as its propagation mechanism: the implementation of email propagation in malware has been evident in a number of subsequent events. For example, the destructive Win32/ExploreZip, and the more recent Win32/NewApt and Win32/MyPics worms.

While the integration of generic and specific techniques within anti-virus products are of tremendous help to users, they present many problems to testers and reviewers. In the next section, we outline in terms of functional specifications what broad features such a product must have in order to be successful in its goal, before examining these specifications point by point from the perspective of the reviewer or tester of antivirus software . The “big picture” functionality of Hybrid products shows that in order to provide a complete solution, all components of the product must be present; that is, simply capturing a sample automatically does not fulfil the Hybrid requirements.

Functional Components of the Hybrid system

Functionally, a Hybrid system must have the following attributes:

- i. The ability to detect an otherwise unknown virus.
- ii. The ability to grant some form of innate immunity to other computers based upon this detection.
- iii. The ability to provide both (i) and (ii) in a time sufficiently short that epidemic spread of the virus is not allowed even for the case of a “network aware” virus.
- iv. Consistent robustness against both viral and non-viral attack; fault tolerant and self-healing with respect to injury, either intentional or unintentional.
- v. Appropriate security, confidentiality and ease of use features that allow the technology to be easily deployable in both the corporate and home setting.

There are several different ways in which it is possible to meet these requirements. In the following paragraphs, we shall examine some of the more common ways in which this functionality may be provided.

i. Unknown virus detection

While an anti-virus system designed to meet the challenges of the future will share similarities with systems of today, there will be some important differences. The system must of course initially find the virus. To that end, software that possesses a combination of techniques for virus detection must be present on the users’ computers. At a minimum, these should consist of one or more of the following:

- ◆ Heuristics based upon sample appearance
- ◆ Simulation of code
- ◆ Post-infection techniques, such as checksumming

ii. Innate immunity generation

Once a new virus is encountered or if no cure is available, the system must offer safe, scaleable, and customizable processes for provision of the cure. While there are several possible methodologies possible in order to achieve this result, the most likely architecture of this type of system is that when a new virus is found, a cure is derived, and a central distribution system sends this cure worldwide, granting innate immunity to other machines, even those which have not encountered the virus.

Some vendors may claim that such a process is not necessary: after all, the heuristics did detect the virus without any innate immunity - is it not the case that the population is already immune? Unfortunately, this is a fallacy, as the following illustrates.

Certain heuristic techniques (like behavior monitors and integrity checkers) operate *post infection*, unlike known virus detection that occurs *before* a virus has an opportunity to infect a machine. Clearly, even if the virus is detected by a post infection heuristic, the machine has had the potential to be damaged by any payload the virus author may have implemented. *Thus, it is far better to provide for pre-infection detection*, and to allow other computers access to known virus detection techniques before they encounter the virus, eliminating possible post-infection damage.

No heuristics which operate pre-infection are perfect; thus in the case of a polymorphic or multipartite virus, it is entirely possible that a virus may be detected generically on some samples and hosts and not others. Another reason for not placing too much reliance upon heuristic detection is that by its very nature it is dynamic.

A typical scenario with the current range of products is that a “new” sample might be detected by the heuristics of a product. Subsequent product updates provide the scanner with a signature with which to accurately identify the same virus. In a sense therefore, heuristic detection is transient in nature. Consider also a product whose heuristics lead to a false positive – subsequent “re-tuning” of the heuristics may well lead to the heuristics of later product versions missing viruses they previously detected.

Pre-infection detection is also important when you consider some of the more complex file infectors, especially those that specifically infect Windows PE (Portable Executable) files. The disinfection of files infected with such viruses is a complex issue. Aside from the fact that some viruses are notorious for corrupting files upon infection (take for example WinNT/Infis [Nikishin, 1999]), there is little certainty in relying upon AV product disinfection to restore such files *exactly*.

iii. Speed of response

As one of the primary benefits of synthesizing known and unknown virus detection is the granting of innate immunity, it is important that any Hybrid system be capable of responding in a timely manner to large outbreaks of many new viruses.

The ability to scale to a large number of transactions is important; specifically, the product functionality should allow for the ability to provide literally up to the minute updates; Architecturally, this may require a hierarchical network design, with globally distributed nodes acting as gateway systems to potentially multiple analysis centers.

iv. Robustness

Speculating on the potential problems with robustness of a hybrid system is difficult without proposing a specific architecture in detail. However, there are some generic lines of attack that appear to be more likely.

There are several layers of an automated system that could be attacked by a virus writer, employing one of several different techniques. First, the system may be attacked directly by a computer virus. In such an attack, the virus author would attempt to defeat the system on the client-side, either by developing viruses that evade heuristic virus detection techniques, or simply entirely disabling the client. While such effects are important to predict and take steps to prevent they are nothing new; of interest here is a macroscopic attack on an entire hybrid system. In the case of a hybrid system which employs smart clients sending information in to a central hub for redistribution, the most likely avenue of attack would seem to be to directly impact feature iii.: speed of response.

v. Security, availability and ease of use

Any process that requires the submission of samples or sample fragments must be secure in the “classical” security sense of the word; that is, it must not detract from the confidentiality, availability and integrity of the information that it handles. Additionally, another important aspect of the system is ease of use; even if the technology is completely in place to provide end to end automation of sample capture, analysis and innate immunity generation, if the system is not easy to configure, it will not achieve its goal.

Testing a “Hybrid” Product: A Proposed Methodology for each Functional Class

Here, we outline some possible tests that can be made against the functional specifications discussed so far. Given the level of expertise in the reviewing world, we will not restate testing methods for virus specific products; rather, we will discuss only those features which are specific to the integration of generic and specific techniques. As we have identified five functionality requirements, we shall break down the methodology into five main sections.

1. Unknown Virus Detection

Testing a product against “unknown” viruses is a difficult task for fairly obvious reasons: an unknown virus must somehow be located and presented to the product for detection.

Some reviewers have attempted to test this functionality by either directly or indirectly writing a new virus, and seeing whether the product is capable of detecting it. While this method certainly meets the requirements of testing a product’s ability upon *guaranteed* new samples, there are potential technical and legal drawbacks approach.

- The test must be carried out on several “new” viruses to gain a statistically meaningful result - a single sample provides little information as to the efficacy of the system.
- The ways in which a hybrid system is capable of detecting new viruses must be understood in order to construct meaningful tests.
- There are potentially legal and ethical problems with either writing or encouraging someone else to write a virus.

A possible solution to this problem for product reviewers could be to construct an additional test-set consisting entirely of viruses that are discovered *after* the product submission deadline. Such viruses may be variants of those for which the product has implemented detection, or completely new.

There are immediate problems with this approach as well. Firstly, for the test to successfully measure the ability of the product to detect a virus it has never seen before, there needs to be a guarantee that the virus is indeed new to the product. Another problem is concerned with time. Sufficient time following the product submission deadline must pass in order for sufficient “new” viruses to be collected. Thus the testing process itself is delayed, and so the time for the review to be published is also delayed. Such a delay must not result in the review itself appearing too dated to be of interest to the end-user.⁴

Developers could provide testers with product versions and signature files from which specific virus detection has been removed. The main problem here is that the tester must rely on the developer to provide this “lobotomized” software, requiring a significant amount of honesty on the part of the vendor. Such a scenario would also be open to severe abuse according to (Howard, 2000).

As noted above, a reviewer could test a product’s heuristics by subjecting the product to viruses released after the submission deadline of the product. The recent creation and implementation of the Dynamic WildList [WildList Organization, 2000] has been delayed due to time constraints. However, efforts are underway which can offer testers a way to assess the virus threat on any given day; testers

⁴ It should be noted that a series of such tests could still provide a series of “snapshots” of the ability of a product to perform over time; thus, such tests should not be ruled out.

should take advantage of the opportunities afforded them by this tracking mechanism (The WildList, 2000).

2. Innate Immunity

Testing of innate immunity generation is comparatively easy if one can find a supply of “new” viruses: a protected computer on the networked system can be “shown” these new viruses, and then the viruses can be introduced to other networked/protected machines. If innate immunity has been granted, the systems that initially do not detect a virus should detect and identify the virus specifically, not generically, and offer to provide disinfection when appropriate, without the need for further analysis or intervention.

3. Speed of Response

While the primary hurdle to be overcome when measuring the response of a hybrid product is obtaining valid test samples, there are other issues that must also be addressed in this space. It is important that the following metrics be applied to simple speed of response tests, in order to explore this space fully:

- While the initial “submission to cure generation” time is of importance to the system, this is by all means not the only response time that is important. It is critically important to measure the time taken for the innate immunity to be redistributed both to other clients, and to interim collection points.
- A key consideration is the priority with which the system deals with incoming samples. Is it possible for the system to prioritize incoming samples? If so, according to what criteria should the samples be prioritized? In the case of a system that relies upon one or more remote analysis centers, it is conceivable that the ability to mark samples with some sense of priority could prevent a submission of “Zoo” samples delaying the processing of actual *In the Wild* samples.

One possible solution is to have “levels” of service for users. This can be increasingly important if the remote analysis center employs manual processes⁵.

4. Robustness

In the case of a viral epidemic, where a new virus is sent to multiple clients, the hybrid system must be capable of, wherever possible, pre-processing samples to remove duplicates and to check whether the sample is now known - that is, it must check that the sample is actually new!

As the dominant protocol in use today is TCP/IP, the system must be designed in such a way that entry points are multi-homed (that is, bound to more than one “backbone” provider) as well as hardened against attack. In the event of such an attack, other unloaded systems/nodes should seamlessly pick up the extra load.

⁵ Human antivirus researchers.

This type of distributed system design relies heavily on the construction of a well-designed architecture that is beyond the scope of this paper.

5. Security, Availability, & ease of use

The entire process of virus prevention should be facilitated at the individual administrator level by an easy-to-configure administration console which governs the system, and which is responsible for audit and control of the various system processes. Detecting viruses is the most important function of the antivirus product – however, it is not the only function that must be fulfilled in order to actually protect the user!

The customization should allow for the administrator to remove any confidential information automatically before a sample is sent; additionally, the administrator should be able to configure the system to require manual approval before a sample is sent, or to send it automatically.

The administrator should be provided with the ability to track the status of all suspected viruses - those which are being examined at an administration level, those which have been sent to an analysis center for processing and those which have been processed.

To facilitate availability, these systems must be capable of offering the most up-to-the-minute updates. Should a cure not be found in those updates, the system needs to quickly and securely seek out the cure from other nodes; if one is not available, it must begin processing the sample as a “new virus”, obtain the cure and make it available for distribution. Obviously as the virus problem grows, it may become necessary to scale up to larger transaction volumes. One effective design which would facilitate such large-scale transaction updates is the active hierarchical network design.

The hybrid system should be capable of being integrated with back office systems that perform tasks such as tracking customer incidents, building new virus definitions, and maintaining a database of virus definitions. Customer incident numbers or identification designators should be assigned consistently so that technical support staff can respond properly to customer calls about the status of a sample that has been submitted; virus definition version numbers must be assigned sequentially so that it is clear that one set of definitions is a superset of previous definitions.

Finally, false positives, in which anti-virus software claims that there is an infection when none is present, should be anticipated by the response system, and all such false positives should be avoided.

Should there be any problems at any step in any of these processes, the system should be capable of deferring problems to human analysts - of notifying the humans immediately that such a problem exists so the problem can be handled expediently.

In terms of confidentiality, there are two basic criteria which must be met: the data sent from the client site must be incapable (or extremely unlikely) of leaking

confidential information to any third party, including the owner of the analysis center. While this criteria is far less important for standard executable files, it is crucial for objects infected with macro viruses, which frequently contain highly privileged data. Any automatic distribution system must be capable of stripping all user data, leaving just a functional virus in its place.

The second criterion requires that information sent between a user site and any transaction center is transmitted securely. Additionally the international nature of communication indicates the use of non-proprietary international standards. Tests of these response mechanisms should include an analysis of the transaction and transport protocols as well as of the cryptographic techniques employed. DES, RSA, MD5 and DSA are recommended cryptographic primitives. While other designs are feasible, HTTP appears to be the most desirable transaction protocol; currently, TCP/IP is the most stable transport protocol, with SSL providing reasonable security functionality.

In terms of availability, the system must be capable of meeting the robustness requirements laid out in section (vi). Additionally, the system must be capable of 100% automation end to end, so that near 100% availability and rapid response times can be guaranteed.

Summary

We have examined generic testing techniques for antivirus products, and discussed some general procedures for testing high-level functionality regardless of implementation. This has allowed us to present a unified checklist for testers.

Given the current state of the art in anti-virus technology however, the most likely implementation of future products is the client-server approach. In this, client machines “capture” suspect samples and send them to one or more central analysis systems that provide ‘known’ virus detection for that virus. For such a system, these initial testing criteria can be considerably simplified, as the following properties must be met:

- Can the system detect new, previously unseen viruses?
- Can the system automatically, without user interaction, “capture” samples of such viruses, or sufficient information about such viruses and automatically send them to an analysis center?
- Can an administrator “vet” samples which are sent? In the case of macro viruses, can confidential information be safely removed from the sample, preventing the leakage of potentially privileged information to a third party?
- Is the submission scheme scalable to deal with the submission of many different viruses/non-viruses at one time?
- Does the analysis center grant “innate” immunity to other computers automatically and seamlessly in a timely manner?
- Is the system easy to manage and deploy in a large corporate environment? Can the software be pre-configured for rollout within a specific environment?

Conclusion

It is inevitable that we will see a gradual integration between generic and specific virus detection techniques over the next 12-18 months. Network-aware viruses such as Melissa have shown that virus-specific techniques are not sufficient to prevent widespread infection by new viruses; similarly, the inherent drawbacks of post-infection generic techniques and pre-infection heuristics make virus specific detection a more attractive way to prevent and remove viruses. Given this situation, we believe that a series of products that provide a hybrid approach to the problem are likely to evolve. Furthermore, tests that are constructed to show the advantages of this approach are likely to benefit the development of these beneficial techniques.

Developing tests that measure the efficacy of new product functionality must not drive the architectural design of products. Rather, we believe that tests should be based upon user requirements, leaving the design of the product (i.e. the method of fulfillment of this user requirement) up to the developer of the software. Taking this approach, we have developed a user-based set of functionality requirements and a method for testing fulfillment of them, which we believe are ready for direct application to the products of today and tomorrow.

References

- Anyware, 2000. http://www.sherine.com/freeware/Antivirus_Tools2.htm
- Checkmark 2000. <http://www.westcoast.com>
- Gordon, S. 1993. *Evaluating the Evaluators*. Virus News International.
- Gordon, S. 1995. *Is a Good Virus Simulator Still a Bad Idea*. Elsevier Press. Elsevier Advanced Technology. Oxford, UK.
- Gordon, S. & Ford, R. 1995. *Real-World Anti-Virus Product Review and Evaluation*. From the Proceedings of the International Virus Prevention Conference. Washington, D. C.
- Gordon, S. & Ford, R. 1996. *Real-World Anti-Virus Product Review and Evaluation: The Current State of Affairs*. From the Proceedings of the 19th National Information Systems Security Conference, Baltimore, Maryland. <http://csrc.nist.gov/nissc/1996/papers/NISSC96/paper019/final.PDF>
- Gordon, S. 1997. *What is Wild*. From the Proceedings of the 20th National Information Systems Security Conference, Baltimore, Maryland. <http://csrc.nist.gov/nissc/1997/proceedings/177.pdf>
- Gordon, S. 2000. <http://www.badguys.org>
- Howard, R. Private Communication. Used with permission.
- ICSA 1999. <http://www.icsa.net/services/consortia/anti-virus>.

- ICSA 2000.
http://www.icsa.net/html/communities/antivirus/certification/certified_products/
- Kephart, J. & White, S. 1993. *Measuring and Modeling Computer Virus Prevalence* From the Proceedings of the 1993 IEEE Computer Society Symposium on Research in Security and Privacy, Oakland, California, May 24-25, 1993; pp.2-14.
- Laine, K. 1993. *The Cult of Anti-Virus Testing*. From the Proceedings of the European Institute for Computer Anti-Virus Research.
- NAI, 2000. <http://www.winmag.com/library/1999/0901/fea0001f.htm>
- Nikishin, A. 1999. *Inside Infis*. Virus Bulletin. pp. 5-6. November.
- PC-Cillin, 2000. <http://www.antivirus.com/corporate/media/1998/pr071598.htm>
- Symantec, 2000a. <http://www.symantec.com/avcenter/y2k/faq.html>
- Symnatec 2000b. http://biz.yahoo.com/prnews/991019/ca_symante_1.html
- Tanner, S. 1993. *A Reader's Guide to Reviews*. Virus News International. November.
- Thunderbyte, 2000.
<http://www.zdnet.com/pcmag/features/utilities98/antivirus/performance.html>
- Trend, 2000. http://www.antivirus.com/reseller_center/hotsheets/smln.htm
- VTC 2000a. <ftp://agn-www.informatik.uni-hamburg.de/pub/texts/tests/pc-av/2000-04/3intro.txt>
- VTC 2000b. <ftp://agn-www.informatik.uni-hamburg.de/pub/texts/tests/pcav/2000-04/0xecsum.txt>
- Whalley, I. 1999. *Melissa – The Little Virus That Could*. Virus Bulletin. pp5-6. May.
- WildList 2000. <http://www.wildlist.org>

^v We would like to thank and acknowledge Dr. Richard Ford for his preliminary work in hybrid system integration & testing, which led to the research presented in this paper.