virus BULLETIN

EXCERPTS FROM VIRUS BULLETIN COMPARATIVE REVIEWS & FEATURE 2004–2005

WINDOWS SERVER 2003 ENTERPRISE X64 VERSION: DEC 05

With 64-bit systems there is a range of hardware available, with operating systems to match. *Athlon 64* processors seem to be the most commonly used with 64-bit operating systems and thus were chosen as a hardware platform. *Windows Server 2003 x64* version was selected as the operating system.

The most recent WildList available at the start of the test period was that from July 2005. Products were dated no later than 31 October 2005.

ESET NOD32 1.1268(20051031)

ItW Overall	100.00%	**Macro**	100.00%
ItW Overall (o/a)	100.00%	**Standard**	100.00%
ItW File	100.00%	**Polymorphic**	100.00%

Dec 2005 VIRUS BULLETIN 100% www.virusbtn.com

Having been tested in a comparative review two months ago and a standalone review in the November 2005 issue of the magazine (see *VB*, November 2005, p.16), no great surprises were expected from *NOD32*. *NOD32* has ZIP archive scanning turned off by default, although W32/Heidi.A is detected by the engine as a special case, accounting for full detection of this virus in the standard test set. In any case detection was at its usual high levels for this product, and *NOD32* easily obtains a VB 100% award for its collection.

RED HAT LINUX 9: APRIL 05

For this comparative review, the test sets were updated to the latest WildList data available on 24 February 2005. This was the December 2004 data. Additions to the WildList this time consisted of a bunch of tedious worms, which were not expected to present any significant challenge for the products.

Eset NOD32 2.03

ItW File	100.00%	**Macro**	100.00%
ItW File (o/a)	100.00%	**Standard**	100.00%
Linux	100.00%	**Polymorphic**	100.00%

April 2005 VIRUS BULLETIN 100% www.virusbtn.com

Having dabbled with kernel objects in the past, the *Eset* developers have now opted for the simpler life and use *Dazuko* for on-access scanning. The last test of this product on *Linux* demonstrated no technical problems but a whole host of different operations were required to install and configure the product. This has been simplified significantly, with one RPM file replacing the trickery that was required previously. The results of scanning were eminently predictable: all files were detected, with a VB 100% award the equally predictable result.

WINDOWS 2003 SERVER: NOV 04

The test sets were based on the most recent version of the (Real-Time) WildList available on 6 October 2004, the deadline for product submission having been 8 October. Three months had passed since the last comparative review, and that was sufficient time for close to 90 new worms to have been added to the In the Wild (ItW) category. The preponderance of all-but-identical worms in, for example, the W32/Sdbot, W32/Rbot, W32/Agobot and W32/Korgo families made replication a particularly mind-numbing process. Although many are packaged in layer upon layer of obfuscating archive, the files themselves are easily recognisable. With this in mind, a bumper crop of VB 100% awards was expected.

Eset NOD32 1.889

ItW Overall	100.00%	**Macro**	100.00%
ItW Overall (o/a)	100.00%	**Standard**	99.82%
ItW File	100.00%	**Polymorphic**	100.00%

Nov 2004 VIRUS BULLETIN 100% www.virusbtn.com

Coming very close to full detection of all samples in all test sets, *NOD32* continues to be entitled to quote its unblemished record of ItW detection on its marketing materials. If a failure in this area does ever occur, I am sure that the printers of *Eset*'s marketing materials will be as happy as *Eset* will be sad. With no incidents of note during testing, I can only congratulate *Eset* on achieving another VB 100% award.

TESTING HEURISTIC DETECTION IN A REAL-WORLD SCENARIO

Andrew Lee
ESET LLC, USA

'*Statistical thinking will one day be as necessary for efficient citizenship as the ability to read and write.*' H.G. Wells.

The results are often published of anti-virus product tests that have no scientific basis, have seriously flawed premises, or demonstrably incorrect methodology. Worse, some tests have all of these faults.

Unfortunately, this is often true of tests conducted by popular computer magazines, whose reviewers seem to think that testing against "a few samples we collected from the Internet and our email" constitutes a valid test of an anti-virus product's detection capabilities. Sadly, it has also been true of some of the tests conducted by recognised AV testing bodies for publication by such magazines – whose interpretation of the results has been wildly misguided.

Reviews in consumer magazines are far more widely read than those in anti-virus industry journals and, in general, their readers are not qualified to understand the (in)significance of the test results. As a consequence, the public's confidence in anti-virus products is at stake.

A classic example, circa 2000, is that of *CNET*'s testing utilising the infamous Rosenthal Utilities (RU). The Rosenthal Utilities generated benign (i.e. non-viral) files, the data part of which contained a portion of a virus. The executable part displayed a message on the screen. It should have been obvious that the detection of any RU-generated file constituted a false positive. However, *CNET* rated the products in the test according to their detection of the RU files. Two years later, *CNET* was not only still making the same mistake [1], but compounding it by altering real viruses such as VBS/Loveletter (i.e. creating new viruses) and trashing products that 'failed' their tests. Eventually, partly due to pressure from the industry [2, 3], *CNET* phased out the use of RU test files.

THE UNDETECTABLES

Particularly misunderstood, and consequently frequently maligned, are the heuristic capabilities of anti-virus products, the testing of which has been woefully inadequate in almost every case this author has seen. This article aims to discuss a scientific basis for real-world testing of heuristics-based anti-virus products, but will begin by describing more general test methodology.

SCIENTIFIC TESTING METHODOLOGY 101

Good testing will aim to prove a hypothesis, will fully define its premises and methodology, and will note any limitations and special considerations.

When comparing products that have different features, it should be stated clearly which features are being tested, and whether the features are common to all products. As far as possible, reviewers should test like against like. Where the testing methodology is likely to have an effect on the product's performance (for instance, testing 'best settings' or default settings), this should be indicated.

Scoring, if any, should reflect only the common features tested – any extra features should either not be scored, or should be scored separately. Any weighting should be clearly defined, and the raw results should be made available. The following are essential to all tests:

- Statistical integrity, using recognised methods.
- Full documentation.
- Presentation of results that arise from scientific analysis of the test data, rather than a subjective view.
  (It is astonishing how often the final 'score' of a product bears little resemblance to the test data).
- The availability of full sample sets to the product manufacturer and independent bodies (the latter of whom must have the ability to verify any results after the fact).

All too often sample selection is seriously flawed, leading to incorrect test results. The worst mistakes include testing against unreplicated (or non-replicable) files, altered files or renamed files (the latter can be an issue, for instance, if scanning by extension is default, and the test is against default settings).

The size of the sample set is also an issue. Testing products against two or three files will prove almost nothing about the detection capabilities of a product – the smaller the sample size, the greater the likely error. However, there is an interesting paradox here. In virus detection terms, the statistically significant portion of the overall sample set (every virus ever written) comprises less than five per cent of that set. The majority of viruses exist only in the 'zoo' collections of the various AV companies and related labs.

For testing purposes those viruses that appear on the WildList are more significant than those that appear in zoo collections. Furthermore, there are two levels to the WildList: the top list, which constitutes viruses that have been reported by two or more WildList reporters, and the supplemental list, where only one reporter has submitted a sample. A sample of a virus with only one report carries only a little more statistical weight than a zoo virus.

There is little agreement on what constitutes a zoo sample set, and therefore it is likely that most testers will use different test sets (there are frequent cases of tests against "files we found in our email"), which introduces biasing error. It is also quite possible, given the huge number of new samples each month, that there are statistically significant amounts of junk in many zoo sets. Amongst the junk is a large amount of 'grey' material. These may be files that are dropped by a virus (for instance a log file) or files that are used as part of an infection sequence. There may be damaged (non-replicative) files, Trojans, jokes, intended viruses, clean files (which includes the non-viral files dropped by a virus), old files that won't run on modern systems, and any amount of other 'noise' [4, 5].

AUTHORIZED REPRINT ©2006 VIRUS BULLETIN LTD

This, and a lack of agreed definitions about what should be detected or reported, means that testing is increasingly difficult to perform correctly – certainly in terms of the statistical implications.

It is as important to know *why* a product failed a particular test, as it is to know that it failed. If the failure was as a result of flawed test design, it should be possible to prove this from the test documentation. The documentation should be written prior to testing, and should define hypotheses and include the full methodology. Any deviation from the documentation in testing should be noted and justified.

Scoring calculations and formulae should be stated and should be demonstrably linked to the test hypotheses. Any score weighting should be explained fully and justified.

Product failures that fall outside of the stated premises should not be factored into the final scoring. For instance, if a product has a stability or installation problem, and neither the test hypotheses nor premises refer to these, the fault should not be counted as a failure. Where the product cannot be tested because of these problems, it should be noted that the product was excluded on this basis, but no failure rate should be given. Failures in individual polymorphic virus sample sets should show as a percentage of that discrete set (usually >1000 replications per sample), not as a percentage of the overall detection rate, or bias will be introduced.

Scored results should be differentiated according to each hypothesis, and if an averaged score is calculated, the method of its calculation should be stated.

DEFINING A TEST SCENARIO

This is not intended to form an exhaustive methodology, nor does it claim to be the best test scenario for testing heuristic product capabilities; rather it will provide a basis upon which a more thorough methodology could be built.

To truly test heuristic detection, the ideal sample set would constitute viruses that are unknown to the product at the time of the test. It is possible to test heuristics against a larger sample set, such as the WildCore, by removing the product's update files (assuming the product works in that way).

A better way would be to 'freeze' a product in time (i.e. not update it for a period), then test it against viruses that emerge in the period between freeze and test. Heuristic capabilities are usually as frequently updated as more conventional detection, so testing against very old (weeks or months) versions of a product will not give a true indication of its capabilities. Ideally, the product on test would be the latest version, without signature updates.

SAMPLE VERIFICATION

The virus sample set must be composed only of verified, replicated samples. The only valid proof that a file is viral is its ability to replicate (ideally to a second generation – but a single replication will usually qualify, with the exception of polymorphic sets).

Using an AV product (or group of products) to determine whether or not something is a virus and perform selection of a sample set is an invalid methodology because neither its error rate, nor its hit rate are statistically determinate. If a test is carried out with un-replicated samples (or assumptions are made about their viability by inference), there will be no scientific basis to the test. Unfortunately there are many cases of tests in which a 'black-box' sample selection is combined with no documentation and no availability of the samples for post-test verification. Especially where sample sets are small, this sort of problem can significantly bias the test results.

PARAMETERS AND HYPOTHESES

Regard should be paid to the default operating mode of the product. For instance, some products have a more aggressive heuristic detection when scanning mail streams (POP3 and SMTP) than in a resident scanner. If this is the case, the most 'real-world' scenario will be to use the correct scan component in its default setting, or a command line version of that scanner with equivalent switches enabled (if available).

Example hypotheses for heuristics testing are as follows:

1. A heuristics-enabled anti-virus scanner will pick up >0% of viruses without the need for updating its signature-based detection.
2. The higher the percentage the better the heuristic analysis can be judged to be.
3. Heuristic analysis detection will not significantly increase the incidence of false positive detection (it may be useful to define this as a percentage, but this is not always necessary).

Example premises:

- Heuristics mitigate a degree of risk of being infected with new viruses.
- A heuristics-enabled scanner will give a better than zero (i.e. better than non-heuristic products) chance of detecting a new or modified virus and protecting against it.

CONTROL TESTS

It is a useful demonstration for each product to be re-tested against the same sample set (e.g. full ItW set) with the full product update at the test date. The purpose of this is to demonstrate whether, under normal operating and updating conditions, detection would have been available against the sample set.

This test has a key statistical implication. The failure rate of the product against the sample set in its fully updated state is an inference statistic of the overall failure rate against the full ItW test set. The failure rate in the heuristic test is *not* an inference statistic against the entire ItW test set, i.e. it is not correct to say that the failure rate (of detection) against the entire parent set (WildCore) would be the same as against the update frozen heuristic test set (unless the full WildCore test set was being tested entirely heuristically – without any updates), because detection should be available for pre freeze ItW samples. Gaining the correct failure rate inference statistics is critical to the soundness of the test result.

AUTHORIZED REPRINT ©2006 VIRUS BULLETIN LTD

Ideally, a full test of the updated product against the ItW sample core would be carried out, but the full update detection test against the post freeze sample set is a sound inference statistic as long as the sample set is statistically significant (i.e. at least 10 per cent of the size of the parent set – again expected error rates should be calculated).

Ideally, a heuristic approach will maintain a zero per cent false positive (FP) rate as well as a zero per cent false negative (FN) rate. The clean sample test will determine a failure rate.

In this case, the hypothesis does not require a zero per cent FP rate, however, a statistically high FP rate is undesirable, and can be used as a measure of misdirected heuristic aggressiveness.

It should be noted that some test scenarios, for instance, pop3 or SMTP traffic, can have a higher suspicion value attached to any executable code that is transferred, and false positive rate is unlikely to have direct negative impact on the system stability or customer reaction – but this is no reason not to test for FP. (A false positive in an on-access [memory resident] scanner can have serious implications, particularly for beleaguered corporate support technicians, and it is usual to have a less aggressive heuristic default [if any] set for this reason.)

Ideally, this control set should be an order of magnitude larger than the viral sample set, preferably an order of magnitude larger than the parent set. In the case of any false alert the file alerted against should be recorded, and copies of the file made available to the product manufacturer.

It should also be noted whether the FP was generated against a specifically crafted test file (some testers use denatured or damaged viruses or files deliberately created to look suspicious), or against a 'normal' file that would exist in a standard system. Ideally, failures against such specifically created suspicious files should be noted as a separate statistic, or simply not recorded; after all, we are discussing real-world scenarios.

An ideal clean file control set will take the form of a statistically valid set of verified clean files found in normal end-user systems.

METHODOLOGY

As mentioned briefly above, there are two tests that can be carried out in terms of heuristic capabilities, and it is important to state which is being undertaken. The first measures the product's capability without update over time, and the second measures its capability against specific new malware as it is released.

The first test is likely to show a significant decrease in detection over time, as new types of virus are encountered and, while it is perhaps an interesting test, the results are largely irrelevant, as a correctly installed product (and therefore its heuristics) will be updated – a fact that this test necessarily ignores. It is unlikely, in any event, that evaluation of heuristics on a product that is more than three months out of date reflects the product's actual capabilities.

In the second (and arguably more valuable) test the update freeze point is important, first because each sample must be new for each product, otherwise the heuristics are not being tested, and secondly, because the product should be the latest available to the public before the outbreak of the malware. Most products update automatically every hour, or at least daily, so the test should not be performed using a product that is significantly older than this (say, not older than 12 hours).

STATISTICAL IMPLICATIONS

Testing heuristics will naturally give some degree of bias because the test set is not fixed. If the test is carried out with a different update freeze, or an expanded sample set, bias will be lesser or greater. For this reason, two identical test scenarios using different freeze points and post freeze sample sets will be likely to produce significantly different results.

Assumptions about the overall performance of a product can only truly be made over a long period of time, with repeat tests. This would provide a mean average of performance. A single update – for example, a day later than the original update freeze (to detect the first virus in a new family) – may skew the result significantly if, as in the case of the Netsky and Bagle families, many variants are subsequently released (a modification of a known virus should be easier to detect heuristically than a totally new sample). A wider diversity of families appearing within the post freeze sample set would reduce the detection rate across the whole product range.

It would be nearly impossible to create a totally sound model for heuristic testing, as it would require a full test of every available post update freeze sample set against every possible update freeze point. Having said that, it is possible to create a model that does give a statistically valid result – if sufficient scientific rigour is applied, and the results are expressed in non-absolute terms.

REFERENCES

[1] A *CNET* review (via techrepublic) demonstrating flawed methodology, http://techrepublic.com.com/5102-6270-1043870.html.

[2] In 2000, members of the AV industry, led by Joe Wells, wrote a public letter to *CNET*, denouncing its testing methodology. See http://www.nod32.com/news/joe_wells.htm.

[3] A response to the May 2002 CNET test can be found at http://www.nod32.com/news/cnet_zdnet.htm.

[4] Bontchev, V., 'Analysis and Maintenance of a Clean Virus Library' available online at http://www.virusbtn.com/old/OtherPapers/VirLib/.

[5] Kaminski, J., 'Malware Evolution As A Cause Of Quality Deterioration Of Anti-Virus Solutions', in U.E. Gattiker (Ed.), *EICAR 2004 Conference CD-rom: Best Paper Proceedings*, Copenhagen: EICAR e.V.

[6] Real Time WildList, http://www.wildlist.org/WildList/RTWL.htm.

AUTHORIZED REPRINT ©2006 VIRUS BULLETIN LTD