Diversity for Security: a Study with Off-The-Shelf AntiVirus Engines

Peter Bishop, Robin Bloomfield, Ilir Gashi, Vladimir Stankovic

Centre for Software Reliability, City University London, London, UK
{pgb, reb, i.gashi, v.stankovic}@csr.city.ac.uk

Abstract

We have previously reported [1] the results of an exploratory analysis of the potential gains in detection capability from using diverse AntiVirus products. The analysis was based on 1599 malware samples collected from a distributed honeypot deployment over a period of 178 days. The malware samples were sent to the signature engines of 32 different AntiVirus products hosted by the VirusTotal service. The analysis suggested significant gains in detection capability from using more than one AntiVirus product in a one-out-of-two intrusion-tolerant setup. In this paper we present new analysis of this dataset to explore the detection gains that can be achieved from using more diversity (i.e. more than two AntiVirus products), how diversity may help to reduce the "at risk time" of a system and a preliminary model-fitting using the hyper-exponential distribution.

1. Introduction

All systems, including those built from off-the-shelf components, need to be sufficiently reliable and secure in delivering the service that is required of them. There are various ways in which this reliability and security can be achieved in practice: use of various validation and verification techniques in the software construction phases, issuance of patches and service releases for the product in operation, as well as the use of software fault/intrusion tolerance techniques. Fault tolerance techniques can range from simple "wrappers" of the software components [2] to the use of diverse software products in a fault-tolerant system [3]. This latter strategy of implementing fault tolerance was historically considered prohibitively expensive, due to the need for developing multiple bespoke software versions. However, the wide proliferation of off-the-shelf software for various applications has made the use of software diversity an affordable option for fault tolerance against either malicious or non-malicious faults.

Intrusion-tolerant architectures that employ diverse intrusion detection systems for detecting malicious behaviour have been proposed in the past [4]. A more recent publication [5] has also detailed an implementation of an AntiVirus platform that makes use of diverse AntiVirus products for malware detection. A similar architecture that uses diverse AntiVirus email scanners has been commercially available for several years [6]. Therefore, architectural solutions for employing diverse detection engines (either IDS or AntiVirus products) are already known and in some cases commercially deployed. Studies that provide empirical evaluation of the effectiveness of diversity for detection of malware and intrusions are, on the other hand, much more scarce.

The following claim is made on the VirusTotal site [7], [8]: "*Currently there is not any solution which provides 100% detection rate for detecting viruses and malware*". Given these limitations of individual AntiVirus engines, designers of security protection systems are interested in at least getting estimates of what are the possible gains in terms of added security that the use of diversity (e.g. diverse AntiVirus products) may bring for their systems.

In this paper we aim to address this research gap. We performed an analysis of the effects of diversity taking advantage of real-world data, namely the information provided by a distributed honeypot deployment, SGNET [9], [10]. We analysed 1599 malware samples collected by the SGNET distributed honeypot deployment over a period of 178 days between February and August 2008. For each malware sample, we studied the evolution of the detection capability of the signature-based component of 32 different AntiVirus products and investigated the impact of diversity on such a capability. Through daily analyses of the same sample using the most up-to-date signature database available for each AV product, we were able to study the evolution of the detection capability over time.

Utilising this dataset, we have reported previously [1] results which analysed the detection capabilities of the different AntiVirus detection engines and potential improvements in detection that can be observed from using *two* diverse AntiVirus detection engines. We observed that some AntiVirus products achieved high detection rates, but none detected all the malware samples in our study. We also found many cases of regression in the detection capability of the engines: cases where an engine would go from detecting the malware on a given date to not detecting the same malware at a later date. When using *two* diverse detection engines we saw significant improvements in the detection capability.

In this paper we present new analysis of our dataset. We have quantified the possible gains in malware detection from using *more than two* diverse engines. We also analysed the dataset in the time dimension to quantify the extent to which the use of diverse AntiVirus engines reduces the "at risk time" of a system. Finally we observe that a hyper-exponential model seems a very good fit for the probability of observing zero-failure- rate systems as we add more diverse AVs.

The evaluation proposed in this paper is defined upon a simplified view, which we consider sufficient to the accomplishment of our goals. We do the following:

- We take into consideration a single type of component appearing in most AntiVirus products, namely the signature-based detection engine.
- We perform the analysis on unambiguous malicious samples, samples that are known to be malicious executable files according to the information provided by the SGNET dataset.
- We consider as a successful detection any alarm message provided by the component. We do not try to diagnose the "correctness" of the generated alarm message.

While the resulting measures may not be representative of the full detection capability achieved by the real-world operation of the various AV products, they provide an interesting analysis of the detection capability of the respective signature-based sub-components under the stated conditions. Also, the purpose of our study is not to rank the individual AntiVirus engines, but to analyse the benefits of diversity that a user may observe from improved detection rates of using more than one AntiVirus product.

For the sake of brevity, in the rest of the paper we will use the short-hand notation AV to refer to the signature-based component of an AntiVirus detection engine.

The rest of this paper is organised as follows: section 2 details the experimental architecture used to collect the data; section 3 details empirical analysis of the benefits of diversity with AntiVirus products; section 4 provides details of a hyper-exponential model which proved to be a very good fit to the probability of having a system with an observed zero failure rate when we increase the number of AVs in a diverse system; section 5 presents a discussion of the results and limitations on the claims we can make about the benefits of diversity; section 6 reviews two recent implementations that employ diverse AntiVirus engines for detecting malware or scanning malicious emails and also discusses other related work; and finally section 7 contains conclusions and provisions for further work.

2. Experimental Setup and Architecture[1]

The construction of meaningful benchmarks for the evaluation of the detection capability of different AntiVirus products is an open debate in the research community. Previous work [11] underlined the challenges in correctly defining the notion of "success" in the detection of a specific malware sample. Also, modern AntiVirus products consist of a complex architecture of different types of detection components, and achieve higher performance by combining together the output of these diverse detection techniques. Since some of these detection techniques are also based on analysing the behavioural characteristics of the inspected samples, it is very difficult to set up a benchmark able to fully assess the detection capability of these complex products.

This work does not aim at being a comprehensive benchmark of the detection capability of different products to the variety of Internet threats. Instead, we focus on a medium-sized sample set composed of a specific class of threats.

The analyzed dataset is composed of 1599 malware samples collected by a real world honeypot deployment, SGNET [9], [10]. SGNET is a distributed honeypot deployment for the observation of server-side code injection attacks. Taking advantage of protocol learning techniques, SGNET is able to fully emulate the attack trace associated with code injection attacks and download malware samples that spread using server-side exploits. By deploying many sensors in different networks of the Internet, SGNET collects in a central dataset a snapshot of the aggregated observations of all its sensors. We use this data as input to our analysis and we build our analysis upon a

[1] This section is similar to section II written in [1]. It is given here to help the reader follow the analysis of the results that succeeds this section.

limited, but realistic dataset with respect to the modern trends for a specific class of malware (i.e. malware associated with code injection attacks).

The SGNET information enrichment framework [11] enriches the information collected by the deployment with additional data sources. Two sources are relevant to this work: the behavioural information provided by Anubis[2] [12], [13] and the detection capability information provided by VirusTotal [7].

Every malware sample collected by the deployment is automatically submitted to Anubis to obtain information of its behaviour once executed on a real Windows system. This information is useful to filter out corrupted samples collected by the deployment, which would not be executable on a real system. Such samples proved to be the cause of ambiguities in the detection capability [11]: it is unclear whether such corrupted samples should or should not be detected since different engines often follow contradicting policies.

The foundations of our analysis are derived from the interaction of SGNET with the VirusTotal service. VirusTotal is a web service that allows the analysis of a given malware sample by the signature-based engines of different AntiVirus vendors[3]. All the engines are kept up-to-date with the latest version of the signatures. Thus, a submission of a malware sample to VirusTotal at a given point in time provides a snapshot on the ability of the different signature-based engines to correctly identify a threat in such samples. It is important to stress that the detection capability evaluation is performed on a subset of the functionalities of the detection solutions provided by the different vendors.

Every time a sample is collected by the SGNET deployment it is automatically submitted for analysis to VirusTotal, and the corresponding result is stored within the SGNET dataset. To get information on the evolution of the detection capability of the engines, each sample, for this dataset, is resubmitted on a daily basis for a period of 30 days.

The dataset generated by the SGNET interaction has some important characteristics that need to be taken into account in the following detection capability evaluation.

Firstly, all the malware taken into consideration have been pushed to the victim as a consequence of a successful hijack of its control flow. We can thus safely consider that all the analyzed samples are a result of a malicious and unsolicited activity.

Secondly, all the considered samples are valid Windows Portable Executable files. All these samples run successfully when executed against a Windows operating system.

Thirdly, the malware samples are differentiated based solely on their content. Thus, the frequent usage of polymorphic techniques (an example of which is given in [14]) in malware propagation is likely to bias the number of malware samples collected. Through polymorphism, a malware modifies its content at every propagation attempt: two instances of the same malware thus appear as different when looking solely at their content.

Finally, interdependence exists between the submission of a sample to VirusTotal and the observed detection capability. The VirusTotal service actively contributes to the AntiVirus community by sharing with all the vendors all the submitted samples resulting in improved detection rates across different AV engines.

3. Exploratory Analysis of AV Diversity

We use the dataset introduced in the previous section to perform a detection capability analysis of 32 different AVs when subjected with the 1599 malware samples collected by the SGNET deployment. Exploiting the submission policy implemented in the SGNET dataset, we have considered for each sample the submissions performed on the 30 days succeeding its download. The input to our analysis can thus be considered as a series of triplets associating together a certain malware sample, an AV product and the identifier of the submission day with respect to the download date $\{Malware_i, AV_j, Day_k\}$. For each of these triplets we have defined a binary score: 0 in case of successful detection, 1 in case of failure. Table 1 shows the aggregated counts of the 0s and 1s for the whole period. As previously explained, we have considered as success the generation of an alert regardless of the nature of the alert itself.

For a number of technical reasons in the interaction of the SGNET dataset and VirusTotal a given malware and an AV are not always associated to 30 triplets. In the observation period we have some missing data since some AVs have not been queried on a given day.

Table 1 - The counts of successful detections and failures for triplets $\{Malware_i, AV_j, Day_k\}$

Value	Count
0 – *no failure / detection*	1,093,977
1 – *failure / no detection*	143,031

[2] http://anubis.iseclab.org/

[3] In our study we evaluated the outputs of 32 AVs.

3.1 Summary of Single AV Results

Table 2 lists the top 10 performing AVs[4] ranked by their observed failure (non-detection) rates[5]. The difference between the failure rate values in the second column and in the fourth column is dependent on the definition of a unique "demand". For the failure rates calculated in the second column, a unique demand is a $\{Malware_i, Day_k\}$ pair, i.e. each malware sent on a different day is treated as unique. Hence the maximum number of demands is the product of the number of distinct malware (1599) and the number of days that an AV product is sent this malware (maximum of 30 days), i.e. 1599 * 30 = 47,970.

For the failure rate calculated in the fourth column, unique demands are the 1599 malware samples. In this case, a successful detection by an AV product AV_j of a given malware sample $Malware_i$ happens when AV_j successfully detects $Malware_i$ on all the dates in which it has inspected this malware.

From the results in Table 2 we can see that there is substantial variability in the detection capability of the top 10 AVs.

Table 2 –Top 10 AVs by their failure (non-detection) rates

For all instances of malware		For all distinct malware	
AV Name	**Failure rate**	**AV Name**	**Failure rate**
AV-7	2.7E-04	AV-7	6.3E-04
AV-16	4.5E-04	AV-17	1.3E-03
AV-17	5.9E-04	AV-6	2.5E-03
AV-32	1.0E-03	AV-26	5.0E-03
AV-26	1.5E-03	AV-21	6.3E-03
AV-2	1.5E-03	AV-15	8.8E-03
AV-6	1.8E-03	AV-22	8.8E-03
AV-30	2.5E-03	AV-16	1.0E-02
AV-22	3.2E-03	AV-19	1.0E-02
AV-21	3.2E-03	AV-23	1.0E-02

Table 3 shows the count of single versions within a given band of failure rate when we calculate their average failure rate for all instances of malware in our dataset (47,970 demands). We can see that 6 AV products are performing very badly for this dataset: they fail for more than 10% of all the demands sent to them. It is inconceivable that any system administrator would choose AVs that have such a bad detection rate to be used in their system. Hence we decided to remove the 6 worst performing AVs from further analysis. The decision was also influenced by our goal to make any results of the benefits of diversity appear fairer. A criticism that can be made if these 6 worst performing AVs are included in the analysis is that improvements in detection capability through the use of diversity will of course be higher if you have such poorly performing individual AVs in the mix. By removing them we make our estimates of the benefits of diversity more conservative.

Table 3 – Counts of single AVs per failure rate band.

Failure rate (f.r.)	Count (and % of the total) of single AVs
failure rate = 0	0 (0%)
1.0E-05 ≤ f. r. <1.0E-04	0 (0%)
1.0E-04 ≤ f. r. <1.0E-03	3 (9.37%)
1.0E-03 ≤ f. r. <1.0E-02	13 (40.63%)
1.0E-02 ≤ f. r. <1.0E-01	10 (31.25%)
1.0E-01 ≤ f. r. <1.0	6 (18.75%)
Total single AVs	**32**

3.2. Results with Diverse AVs

3.2.1 Overall Summary

We now look at the benefits that using more than one AV may bring in terms of detection behaviour. The analysis is based on the top 26 performing individual AV products. The observed failure rates are calculated from a maximum of 47,970 demands, as explained in the previous section.

Table 4 gives the results for all the possible distinct *1-out-of-n* systems that can be built from these 26 AV products.

Table 4 – Counts of different 1-out-of-n systems of AVs per failure rate band. Abbreviations: *f.r. - failure* rate;

Bands	*f. r. = 0*	*1.0E-05 ≤f. r.< 1.0E-04*	*1.0E-04 ≤f. r.< 1.0E-03*	*1.0E-03 ≤f. r.< 1.0E-02*	*Total number of systems*	*Proportion of systems with perfect detection*
1oo2	80	40	74	131	325	0.246154
1oo3	1,353	395	528	324	2,600	0.520385
1oo4	10,985	1,617	1,882	466	14,950	0.734783
1oo5	57,033	4,161	4,142	444	65,780	0.867026
1oo6	216,199	7,333	6,382	316	230,230	0.939057
1oo7	641,030	9,227	7,383	160	657,800	0.974506
1oo8	1,547,183	8,506	6,532	54	1,562,275	0.990340
1oo9	3,114,327	5,817	4,395	11	3,124,550	0.996728
1oo10	5,306,587	2,951	2,196	1	5,311,735	0.999031
1oo11	7,724,287	1,082	791	0	7,726,160	0.999758
1oo12	9,657,234	272	194	0	9,657,700	0.999952
1oo13	10,400,529	42	29	0	10,400,600	0.999993
1oo14	9,657,695	3	2	0	9,657,700	0.999999
1oo15	7,726,160	0	0	0	7,726,160	1

The total number of systems in each configuration (the penultimate, right-most column of Table 4) is

[4] The AV names have been anonymised to prevent concerns deriving from the comparison of commercial products.

[5] Any mention of "failure rates" in this section refers to the observed (empirical) failure rates calculated from our dataset.

obtained using the combinatorial formula ${}^{n}C_{r}$. For example, the total number of 1-out-of-2 (1oo2) systems that you can create from 26 different AVs is ${}^{26}C_{2}$ = 325, and so on. For each of these distinct systems we calculate the average failure rate for all instances of malware sent to them. The table only goes as far as 1-out-of-15 because we had no single demand that caused 15 different AVs to fail simultaneously. Hence from 1oo15 onwards we observe perfect detection with our dataset.

The first column of the results (f.r. = 0) shows the numbers of systems of each configuration that perfectly detected all the malware on all the dates that they inspected them. We had no single AV that was in this category (as evidenced in Table 3), but this number grows substantially as the number of AVs in a diverse configuration increases (you can see its proportional growth to all the systems in a particular configuration on the right-most column of the table). The second column (1.0E-05 ≤ f. r. < 1.0E-04) shows the number of systems whose average failure rate for all instances of malware is between 10^{-4} and 10^{-5}. The worst case failure rate band for any diverse setup is 10^{-2} and 10^{-3}. In Table 3 we saw that the worst case failure rate band, even after filtering away the 6 worst single AVs, for any single AV is 10^{-1} and 10^{-2}. Hence, for this dataset, we can interpret this result as "*the worst diverse configuration brings an order of magnitude improvement over the worst single AV product*".

3.2.2 Visualisation of the Results

In this sub-section we will show some graphs to help us visualise and better interpret the data in Table 4 (the 1oo15 results are not shown on the graphs since there are no failures for this configuration).

Figure 1 directly visualises the numbers shown in Table 4.

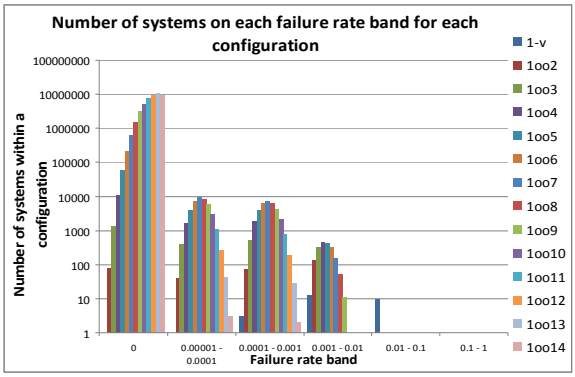

Fig. 1 – Number of systems in each failure rate band for each configuration

Note that the y-axis is shown in an exponential scale. We can clearly observe the shift of mass to the left (i.e. towards the 0 failure rate) for configurations with higher level of diversity.

Figure 2, shows the cumulative distribution function (cdf) of the average failure rate for each configuration. This graph helps us visualise the net average gain in detection capability at each failure rate band when adding extra layers of diversity (i.e. additional diverse AVs).

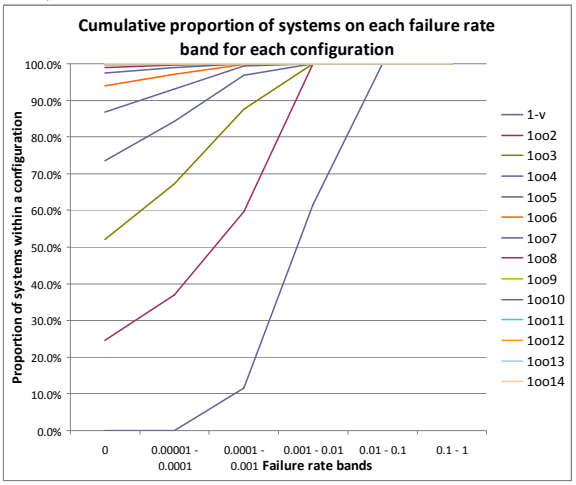

Fig. 2 – Cumulative proportions of systems in each failure rate band for each configuration

3.3. "Diversity over Time" Analysis

As we stated before, each malware sample was sent to a given AV product on several consecutive days after it was first uncovered in the SGNET honeypots. This period was a maximum of 30 days in our dataset. By their very nature signature-based AVs should eventually detect all the malware, provided the vendors write the appropriate signatures to detect them. However the response time for writing these signatures can differ significantly. Hence we analysed to what extent the use of diverse AVs may help a system reduce its "time at risk", i.e. the time in which an AV is exposed to a known malware for which a signature is yet to be defined by a given AV. By using more than one AV the system is only exposed for the time it takes the fastest of the vendors to define the signature.

Figure 3 contains contour plot of the entire dataset of the 26 AVs in our study. The x-axis lists the 26 AVs ordered from left to right (the highest detection rate to the one with the lowest). The y-axis contains a listing of all 1599 malware. The z-axis values are given by the intensity of the colour on the plot, which is given by the legend on the side of the plot. The black coloured lines are associated with malware that were never seen by a given AV (i.e. the missing data): in our dataset these are associated with just one AV product. The

white lines are associated with malware which were always detected by given AV product. The values 1-30 in the z-axis, represented by colours ranging from intense green to light pink, are associated with the number of days that a given AV failed to detect a given malware in our study, but which eventually were detected. The red lines represent the malware which, in our collection period, were never detected by a given AV product.

The graph shows clear evidence of diversity, which explains the results we have observed so far.

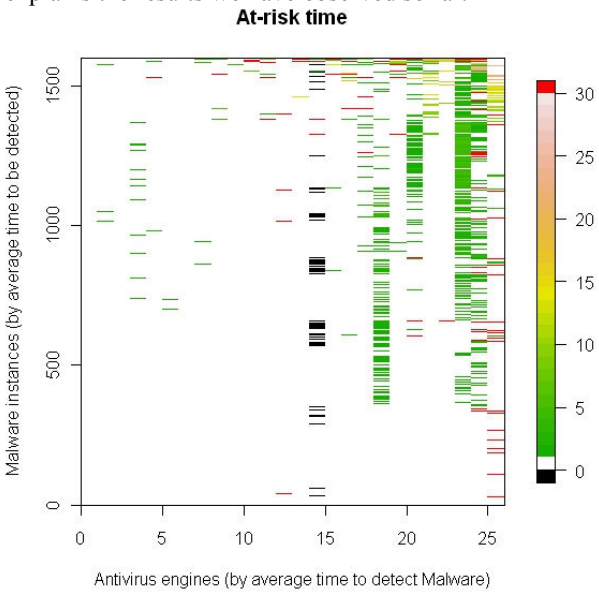

Fig. 3 – The "at risk time" in number of days for each of the 26 AVs on each of the 1599 malware samples in our dataset

We now look in more detail at the most "difficult" 10% of malware (i.e. the ones which were most often missed overall by the AV products). This is shown in Figure 4. Note that this not a zoom of the top-most 10 % of the Figure 3, because the ordering is done over the malware. The ordering is bottom-to-top from the most difficult to detect to the least difficult (within this 10% of malware).

If we look at the graph along the x-axis it is clear again that there are infrequent cases of a line (i.e. a particular malware) running across the different AVs with the same colour. This is evidence that even for malware which on average the different AVs are finding difficult to detect, there is considerable variation on which ones they find difficult (i.e. a malware which an AV finds difficult another one does not, and vice versa) and when they do fail, their at risk time does vary. Hence it is clear that using diverse AVs, even in cases when different AVs fail to detect the malware, can have an impact in reducing the "at risk time" of a system: the time the system is at risk is the minimum time it takes any of the diverse AVs employed to detect the malware.

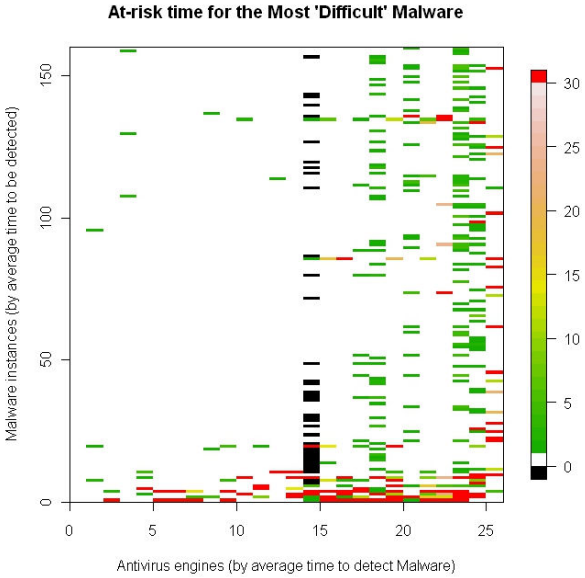

Fig. 4 – The "at risk time" in number of days for each of the 26 AVs when considering the 160 most "difficult" Malware samples.

Another viewpoint of the time dimension analysis is given in Figure 5. For each AV we looked at the time (in number of days) it took it to detect a given malware sample. To make the graph more readable we categorised these times into: always detected (shown on the bars filled in white colour with black borders), never detected (shown in red fill colour), no data (in black fill colour), and then 5 day groupings (1-5 days, 6-10 days etc.). Each bar shows the proportion of malware that are in any of these categories. We are only showing the top 45% in the y-axis, as all AVs always detected at least 55% of all the malware samples sent to them.

Figure 5 allows us to look more clearly at the empirical distribution of the time it takes each AV to detect the malware samples in our study. Along the x-axis the AVs are ordered left-to-right from the AV with the lowest time at risk (AV-7) to the highest (AV-5).

The distribution of detection times may have a bearing on the choice of AVs that a given user may wish to make for a given system. Rather than choosing the "best on average", users may choose the AVs that do not have too many undetected malware (shown in red colour in the bars of the graph in Figure 5). For example, even though AV-23 has a better (shorter) average "at risk time" than AV-18, it is evident from Figure 5 that AV-18 has a shorter red coloured section of the bar compared with AV18. This means AV-18 failed to detect, in our study, a smaller number of

malware than AV-23, even though its total at risk time was worse (higher).

From the viewpoint of diversity analysis, the choice of which set of AVs to choose is therefore not only influenced by the highest detection rates (diversity in "space") but also the time it takes the different AVs to detect malware (diversity in "time"). Different AV vendors may have different policies on deciding which malware they prioritise for a signature definition. Therefore the needs of an individual end-user for whom a definition of a signature for a particular malware sample is of high importance, may mismatch with the AV vendor's prioritisation of that malware. So use of diverse AVs helps avoid this "vendor lock-in". Selection of diverse AVs to optimise the diversity in time aspect requires we choose AVs from vendors that exhibit diverse policies at prioritising signature definitions for different malware.

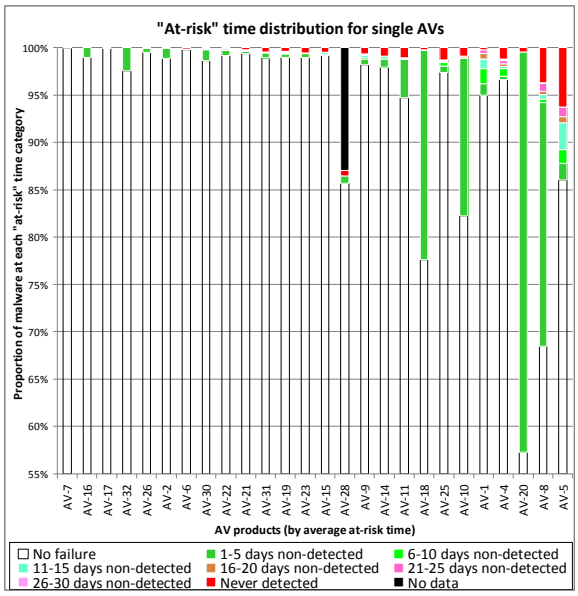

Fig. 5 – "At risk time" distribution for single AVs

4. A Hyper-Exponential Model for the Reduction of Systems that Fail When Adding Diversity

An interesting characteristic of the data shown in Table 4 is the asymptotic reduction in the proportion of systems that fail on any of the malware. A simple exponential model of the decrease with the number of AVs (N) in the diverse system would estimate the proportion as:

$$P(faulty \mid N) = p^N$$

where p is the probability that a single AV fails to detect all the malware.

In practice this proved to be a very poor fit to the observed proportions. However an empirically derived hyper-exponential model of the form:

$$P(faulty \mid N) = (p^N)^{N/2}$$

proved to be a remarkably good fit to the observed data as shown in the Figure 6 below.

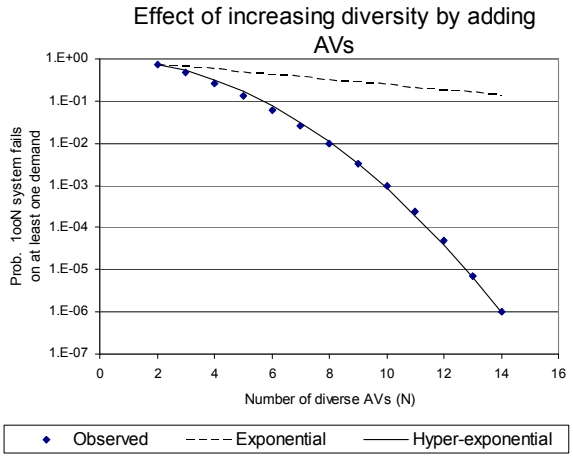

Fig. 6 – Effect of increasing diversity by adding AVs

The results show that the proportion of faulty systems decreases far more rapidly than a simple exponential distribution would indicate. For example, increasing N from 8 to 10 decreases the chance of having a 1ooN system that fails by an order of magnitude.

5. Discussion

5.1 Confidence on the Best Case Observed Failure Rate

Any claim on the best case (i.e. lowest) failure rate we can hope for, using this dataset, will be in the region of 10^{-4} to 10^{-5}. This is because we have a maximum of 47,970 demands sent to any given AV (as we explained earlier). This is the upper bound on the observed failure rate. To make claims about lower failure rates (i.e. smaller than 10^{-5}) we need more data – or, in other words, we must do more testing. The theoretical background behind these intuitive results is given in [15][6].

[6] Section 3 of [15] contains a discussion of the limits on claims we can make about reliability when statistical testing

But the fact that we are seeing only, for instance, 5 out of almost 10 million different combinations of 1oo14 systems fail on this dataset is telling us something about the improvements in detection rates from using diversity (cf. Table 4), even if we cannot claim a failure rate lower than 10^{-5}. We may interpret the 0.999999 proportion of 1oo14 systems with a perfect failure rate as a *99.9999% confidence* we have in the claim that, for this dataset, "*the average failure rate of a 1oo14 diverse AV system constructed from randomly selecting 14 AVs from a pool of 26 AV products, will be no worse than ~ 4.7 * 10^{-5}*". This may or may not persuade a given system administrator to take on the additional cost of buying this extra security: this will clearly depend on their particular requirements. But in some cases, especially regulated industries, there may be a legal requirement to show actually achieved failure rates with a given level of confidence. It may not be feasible to demonstrate those failure rates with the associated confidence for single AV products without extensive testing.

The claim above is limited by the level of diversity we can expect between various diverse configurations. We "only" have 26 AV products (even though this may represent a large chunk of the available commercial solutions available). Hence, for instance, even though we have over 65,000 different combinations of 1oo5 systems using 26 AVs, the level of diversity that exists between these different 1oo5 systems may be limited in some cases. For example, a 1oo5 system which consists of AV1, AV2, AV3, AV4 and AV5, is not that much different from a 1oo5 system consisting of AV1, AV2, AV3, AV4 and AV6. Hence care must be taken from generalising the level of confidence from these large numbers without considerations of these subtle concepts on the sample space of possible systems.

5.2 Discussion of the Hyper-Exponential Model

There is no immediate explanation for the surprisingly good fit of the 1ooN system detection capability to the hyper-exponential model, but we suspect it is related to the Central Limit Theorem where individual variations are smoothed out to produce a normal distribution (for a sum of distributions), or a log-normal distribution (for the product of distributions). In our case, the hyper-exponential distribution could be intermediate between these two extremes. Furthermore, it might be that the asymptotic nature of exceedances and extreme values can be used to justify a general hyper exponential or Gumbel distribution. This is an area that requires more investigation.

If the hyper-exponential distribution is shown to be generic, it would be a useful means for predicting the expected detection rates for a large 1ooN AV system based on measurements made with simpler 1oo2 or 1oo3 configurations.

6. Related Work

6.1 Architectures that Utilise Diverse AntiVirus Products

In this paper we have presented the potential gains in detection capability that can be achieved by using diverse AVs. Security professionals may be interested in the architectures that enable the use of diverse AV products.

The following publication [5] has provided an initial implementation of an architecture called Cloud-AV, which utilises multiple diverse AntiVirus products. The Cloud-AV architecture is based on the client-server paradigm. Each host machine in a network runs a host service which monitors the host and forwards suspicious files to a centralised network service. This centralised service uses a set of diverse AntiVirus products to examine the file, and based on the adopted security policy makes a decision regarding maliciousness of the file. This decision is then forwarded to the host. To improve performance, the host service adds the decision to its local repository of previously analysed files. Hence, subsequent encounters of the same file by the host will be decided locally. The implementation detailed in [5] handles executable files only. A study with a deployment of the Cloud-AV implementation in a university network over a six month period is given in [5]. For the executable files observed in the study, the network overhead and the time needed for an AntiVirus engine to make a decision are relatively low. This is because the processes running on the local host, during the observation period, could make a decision on the maliciousness of the file in more than 99% of the cases that they had to examine a file. The authors acknowledge that the performance penalties might be significantly higher if the types of files that are examined increases as well as if the number of new files that are observed on the host is high (since the host will need to forward the files for examination to the network service more often).

Another implementation [6] is a commercial solution for e-mail scanning which utilises diverse AntiVirus engines.

---

reveals no failures. Both classical and Bayesian inference methods are described.

6.2. Other Related Empirical Work with AV Products

Empirical analyses of the benefits of diversity with diverse AV products are scarce. CloudAV [5] and our previous paper [1] are the only examples we know of published research.

Studies which perform analysis of the detection capabilities and rank various AV products are, on the other hand, much more common. A recent publication [8] reports results on ranking of several AV products and also contains an interesting analysis of "at risk time" for single AV products. Several other sites[7] also report rankings and comparisons of AV products, though one must be careful to look at the criteria used for comparison and what exactly was the "system under test" to compare the results of different reports.

7. Conclusions

The analysis proposed in this work is an assessment of the practical impacts of the application of diversity in a real world scenario based on realistic data generated by a distributed honeypot deployment. As shown in [11], the comprehensive performance evaluation of AntiVirus engines is an extremely challenging, if not impossible, problem. This work does not aim at providing a solution to this challenge, but builds upon it to clearly define the limits of validity of its measurements.

The performance analysis of the signature-based components showed a considerable variability in their ability to correctly detect the samples considered in the dataset. Also, despite the generally high detection rate of the detection engines, none of them achieved 100% detection rate. The detection failures were both due to the lack of knowledge of a given malware at the time in which the samples were first detected, but also due to regressions in the ability to detect previously known samples as a consequence, possibly, of the deletion of some signatures.

The diverse performance of the detectors justified the exploitation of diversity to improve the detection performance. We calculated the failure rates of all the possible diverse systems in various 1-out-of-n (with n between 2 and 26) setups that can be obtained from the best 26 individual AVs in our dataset (we discarded the bottom 6 AVs from our initial set of 32 AV products, as they exhibited extremely poor detection capability). The main results can be summarised as follows:

- Considerable improvements in detection rates can be gained from employing diverse AVs;
- No single AV product detected all the malware in our study, but almost 25% of all the diverse pairs, and over 50% of all triplets successfully detected all the malware;
- In our dataset, no malware causes more than 14 different AVs to fail on any given date. Hence we get perfect detection rates, with this dataset, by using 15 diverse AVs;
- We observe a law of diminishing returns in the proportion of systems that successfully detect all the malware as we move from 1-out-of-8 diverse configurations to more diverse configurations;
- Significant potential gains in reducing the "at risk time" of a system from employing diverse AVs: even in cases where AVs fail to detect a malware, there is diversity in the time it takes different vendors to successfully define a signature for the malware and detect it;
- The analysis of "at risk time" is a novel contribution compared with traditional analysis of benefits of diversity for reliability: analysis and modelling of diversity has usually been in terms of demands (i.e. in space) and the time dimension was not usually considered;
- An empirically derived hyper-exponential model proved to be a remarkably good fit to the proportion of systems in each diverse setup that had a zero failure rate. We plan to do further analysis with new datasets and if the hyper-exponential distribution is shown to be generic, it would be a useful means for predicting the expected detection rates for a large 1ooN AV system based on measurements made with simpler 1oo2 or 1oo3 configurations.

There are several provisions for further work:

- As we stated in the introduction, there are many difficulties with constructing meaningful benchmarks for the evaluation of the detection capability of different AntiVirus products (see [11] for a more elaborate discussion). Modern AntiVirus products comprise a complex architecture of different types of detection components, and achieve higher detection capability by combining together the output of these diverse detection techniques. Since some of these detection techniques are also based on analysing the behavioural characteristics of the inspected samples, it is very difficult to setup a benchmark capable to fully assess the detection capability of these complex components. In our study we have concentrated on one specific part of these products, namely their signature-based detection engine.

[7] http://av-comparatives.org/, http://av-test.org/, http://www.virusbtn.com/index

Further studies are needed to test the detection capabilities of these products in full, including their sensitivities to false positives (whatever the definition of a false positive may be for a given setup);

- Studying the detection capability with different categories of malicious files. In our study we have concentrated on malicious executable files only. Further studies are needed to check the detection capability for other types of files e.g. document files, media files etc;
- Analysis of the benefits of diversity for "majority voting" setups, such as 2oo3 ("two-out-of-three") diverse configurations for example. Since our dataset contained confirmed malware samples, we thought detection capability is the most appropriate aspect to study: a system should prevent a malware from executing if at least one AV has detected it as being malicious (hence our emphasis on 1-out-of-n setups). But in cases where false positives become an issue, using majority voting may help curtail the false positive rate;
- More extensive exploratory modelling and modelling for prediction. The results with the hyper-exponential distribution were a pleasant surprise, but more extensive analysis with new datasets is required to make more general conclusions. Extensions of current methods for modelling diversity to incorporate the time dimension are also needed.

Acknowledgments

This work has been supported by the European Union FP 6 via the "Resilience for Survivability in Information Society Technologies" (ReSIST) Network of Excellence (contract IST-4-026764-NOE), FP 7 via the project FP7-ICT-216026-WOMBAT, and a City University Strategic Development Fund grant.

The initial research [1] on these results was done in collaboration with Corrado Leita and Olivier Thonnard, who are now with Symantec Research.

We would like to thank our colleagues at CSR for fruitful discussions of the research presented, and especially Robert Stroud who reviewed an earlier version of this paper.

References

1. Gashi, I., et al. *An Experimental Study of Diversity with Off-the-Shelf AntiVirus Engines*. in *Proceedings of the 8th IEEE Int. Symp. on Network Computing and Applications (NCA)*, p. 4-11, 2009.
2. van der Meulen, M.J.P., et al. *Protective Wrapping of Off-the-Shelf Components*. in *the 4th Int. Conf. on COTS-Based Software Systems (ICCBSS)*. Bilbao, Spain, p. 168-177, 2005.
3. Strigini, L., *Fault Tolerance Against Design Faults*, in *Dependable Computing Systems: Paradigms, Performance Issues, and Applications*, H. Diab and A. Zomaya, Editors, J. Wiley & Sons. p. 213-241, 2005.
4. Reynolds, J., et al. *The Design and Implementation of an Intrusion Tolerant System*. in *32nd IEEE Int. Conf. on Dependable Systems and Networks (DSN)*. Washington, D.C., USA, p. 285-292, 2002.
5. Oberheide, J., E. Cooke, and F. Jahanian. *CloudAV: N-Version Antivirus in the Network Cloud*. in *Proc. of the 17th USENIX Security Symposium*, p. 91–106, 2008.
6. GFi. *GFiMailDefence Suite*, last checked 2011, http://www.gfi.com/maildefense/.
7. VirusTotal. *VirusTotal - A Service for Analysing Suspicious Files*, last checked 2011, http://www.virustotal.com/sobre.html.
8. Sukwong, O., H.S. Kim, and J.C. Hoe, *Commercial Antivirus Software Effectiveness: An Empirical Study*. IEEE Computer, 2011. **44**(3): p. 63-70.
9. Leita, C. and M. Dacier. *SGNET: Implementation Insights*. in *IEEE/IFIP Network Operations and Management Symposium (NOMS)*. Salvador da Bahia, Brazil, p. 1075-1078, 2008.
10. Leita, C., et al. *Large Scale Malware Collection: Lessons Learned*. in *Workshop on Sharing Field Data and Experiment Measurements on Resilience of Distributed Computing Systems, 27th Int. Symp. on Reliable Distributed Systems (SRDS)*. Napoli, Italy, 2008.
11. Leita, C., *SGNET: Automated Protocol Learning for the Observation of Malicious Threats*, in *Institute EURECOM*. 2008, University of Nice: Nice, France.
12. Bayer, U., *TTAnalyze: A Tool for Analyzing Malware*, in *Information Systems Institute*. 2005, Technical University of Vienna: Vienna, Austria. p. 86.
13. Bayer, U., et al., *Dynamic Analysis of Malicious Code*. Journal in Computer Virology, 2006. **2**(1): p. 67–77.
14. F-Secure. *Malware Information Pages: Allaple.A*, last checked 03 June 2011, http://www.f-secure.com/v-descs/allaple_a.shtml.
15. Littlewood, B. and L. Strigini, *Validation of ultrahigh dependability for software-based systems*. Commun. ACM, 1993. **36**(11): p. 69-80.