symantec™ Security Response

Reputation-based Security

An Analysis of Real World Effectiveness

Zulfikar Ramzan, Vijay Seshadri, and Carey Nachenberg

Contents

Abstract

In September of 2009, Symantec released its first reputation based security offering as a part of its consumer security products. Our experiences over the past year have provided us with valuable insight into potential challenges and pitfalls of deploying a widespread file reputation system. They've given us insight into the types of threats that can be effectively detected using a reputation system and they have helped us to understand how to adjust the system to maintain its overall effectiveness over time.

This paper presents an analysis of the real world effectiveness of reputation based security in detecting new malware. First, it provides an overview of the concept and how it is implemented in the overall context of our security products. We then present techniques used to measure the effectiveness of this technology (true positive and false positive rates) as well as the technical challenges we faced in evaluating this brand new anti malware detection approach. The paper concludes by summarizing the overall impact of reputation based security on the malware threat space and, more generally, on the AV industry.

Reputation-based Security Overview

Attackers have shifted their attack strategy over the past decade. In the past, they mass distributed a small number of highly prevalent threats to thousands or millions of users. Today, they micro distribute millions of distinct, mutated threats, infecting each user with a new variant with its own distinct (and likely unknown) fingerprint. We are seeing that over 75% of malware detected by reputation technology is in less than

50 Symantec users. This paradigm shift has proved a huge challenge for traditional fingerprint and heuristic-based antivirus techniques. Why? First, given the rarity of each threat, many are likely to never be discovered, and if they're not discovered, they can't be fingerprinted. Second, it is inefficient to release a fingerprint to hundreds of millions of users when that fingerprint might only protect one or two users across the globe. With over 600,000 new variants being created per day (Symantec received 240 million unique threat hashes last year from protected customer machines), it is unfeasible to create, test, and distribute the volume of signatures necessary to address the problem.

Four years ago, Symantec embarked on an ambitious plan to reinvent antivirus; we have developed an entirely new, reputation based approach that accurately classifies files based on their distribution (or lack thereof) across our huge user base. Over 75 million participating users (as of June 2010) send us anonymous, real time telemetry data about the applications they use. In addition, Symantec can also leverage data from Symantec's Global Intelligence Network, various web crawlers, other Symantec product offerings that have data submissions capabilities, Symantec's Security Response organization (e.g., for data about malicious software), and legitimate software vendors who provide application instances to Symantec.

This data is incorporated into a massive model that represents executable files (and associated file meta data), anonymous users, and the linkages between the two. We then run reputation algorithms on this model to compute highly accurate reputation ratings on every single file, both good and bad, known to our participating users. Our system then delivers the prevalence, age and reputation score to all of our client and gateway products to help improve their protection.

Such an approach is not only effective against popular malware, but can also identify even the most arcane threats — even those affecting just a handful of users across the entire Internet. And our system doesn't just identify and classify bad files. It computes a reputation score for every single executable file, both the bad ones and the good ones, used by every participating Symantec user.

Symantec released the first blocking version of reputation based security in its Norton consumer security products in September of 2009, though some form of reputation based security has been in the products for the last few years (and a data gathering phase was embarked upon even earlier).

How Do We Use Reputation Ratings?

Use case #1: Download Insight

The most visible aspect of the reputation based security feature is the Download Insight feature (DI). In our consumer products, DI intercepts every new PE file on download from the Internet and queries the Symantec reputation cloud for a rating. Based on ratings received from the cloud, DI takes one of three different actions. If the file has developed a bad reputation, then it is blocked outright. If the file has developed a good reputation, the file is allowed to run. Finally, if a file is still developing its reputation and its safety is unknown, the user is warned that the file is unproven; the user can then decide, based on their tolerance for risk, whether or not they want to use the file.

Given most threats are downloaded from the Internet through browsers, we believe that DI is very effective in preventing new infections. In this model, when executable content gets downloaded through the browser, DI intercepts and performs a reputation scan. Given that reputation-based security is very effective in detecting threats that target few users, we are seeing a very high detection rate for new threat infections.

In addition to reputation ratings, DI also provides metadata about each file to the user when asking them how to address an unknown reputation file: it's prevalence, age (when it was first discovered by a Symantec user), its origin, etc. We believe this is very useful information for the user to make a decision on whether to execute the file or not. For example, a risk averse user might not want to run a file if is a few hours old and only a few users have it, where as an expert user might consider the origin of the file and run it anyway.

In Enterprise settings, the administrator will be able to author file blocking policies that take into account prevalence and age in conjunction with reputation. This approach enables the administrator to deploy blocking policies that match the risk tolerance for the establishment or for a given division. For example, a very large bank might decide that it will not permit any file to be introduced to their network unless it has a high reputation, has at least 1,000 users, and is at least a month old. In contrast, a small business might be less risk averse and allow its users to use any software with at least 100 users and a medium reputation. These types of policies can be crafted on a per user (or per group) basis. For example, a user who gets perpetually infected could be assigned more restrictive settings, whereas a highly tech savvy user with no prior infection history can be given more lax settings. A similar consideration can apply based on a person's role within the organization. A highly value machine (such as that of a CEO) can have an appropriate policy associated with it. Furthermore, an administrator may choose to amend policies based on other criteria. For example, policies can become more restrictive in the period following an outbreak or in a particularly sensitive period for the organization (e.g., in the period surrounding merger and acquisition activity). Because a file's reputation, its prevalence and its age can yield multiple measures of trustworthiness, these attributes lend themselves to crafting policies with far richer expressiveness than schemes in which a binary value (malicious vs. benign) is attached to a file.

Use case #2: Corroboration Engine

In addition to detecting threats on download, the reputation-based security system also acts as an orthogonal corroboration engine for Symantec's heuristic engines. Traditional heuristics and behavior blocking systems base their detection upon the static/behavioral attributes of a given executable in the context of a single machine. For example, traditional systems don't know whether a program's behavior is similar on other computers. Likewise, they don't know whether the program being evaluated has widespread distribution or is literally only on a single machine, worldwide. In other words, traditional anti-malware technologies take on a myopic view of the world by examining a software application on only one machine at one point in time. In contrast, a reputation-based security system offers a worldview for every program; it knows a program's distribution and adoption patterns across the entire user base. Since this data is complementary, it can be used to improve the accuracy of traditional heuristics and behavior blocking engines. Symantec's traditional technologies combine features they collect about an application on a particular machine with features collected from Symantec's reputation system. The combined information is used to establish a (more accurate) verdict for a software application.

Moreover, this data fusion approach permits the possibility of very loose heuristics and behavioral detections coupled with a broader reputation safety net. This combination leads to to increased detection (or true positive) rates while managing the risk associated with false positives.

In addition, reputation based techniques can help alleviate high profile false positives. For example, we can choose to employ specific heuristics or behavioral detections only on files that are on fewer than N machines. The expectation is that for (malicious) applications with prevalence greater than N, we are better off relying on traditional signatures. Going further, the architecture also enables centralized mitigation of newly discovered false positives. For example, if a signature in our traditional anti virus engine exhibits a false positive on an application, we can effectively manually assign the file a very high reputation score on the back end. Because both pieces of information are considered in concert, the sterling reputation score assigned on the back end can effectively trump the conviction attempt by a signature on the anti virus engine. By leveraging the "cloud" we can short circuit the more expensive patch distribution process that the industry normally uses to handle such situations.

Use case #3: Enhanced Performance

While the previous two sections illustrate how a reputation based system improves security, we have found that a reputation system can also be used to substantially improve performance. A reputation-based system doesn't just track bad files — it has ratings for all files, both good and bad. One possible use for this data is to identify extremely high reputation good files on protected computers. Once these high reputation files have been identified, they can be excluded from further antivirus scanning and behavioral profiling. As one might imagine, the vast majority of software applications on a typical end user system are well known. Today our products are able to identify, as highly trusted, roughly 80 – 90% of actively running applications on a typical user's system. Once our products identify such a trusted application on a customer's machine, they never scan it again (unless a revocation occurs or the file's contents are modified). In contrast, traditional antivirus products typically scan (or behaviorally monitor) every program every time virus definitions are updated. These reputation based exclusions not only improve performance but allow the scanning engines to spend more resources analyzing the smaller set of less trusted files (which are more likely to be malware), hence protecting the machine better.

Measuring Effectiveness

Cumulative vs. Interactive Effectiveness

One of the big lessons learned over the last year is how to best measure the efficacy of a reputation system. Originally, we figured that the traditional TP/FP measurements used for years with antivirus software would be sufficient. We call this approach the Cumulative Approach, and describe it below. However, during the past year, we identified a second (we think better) way to measure the effectiveness of a reputation system: the Interactive-based approach.

In the traditional Cumulative Approach, we collect a sampling of thousands of known, unique good and bad files, as distinguished by their SHA2 hash, and then measure how many of these files are classified correctly and how many are classified incorrectly by our reputation system based on its latest, most up-to-date cumulative telemetry about these files. For example, we might sample 10,000 known, unique good files and 10,000 known, unique bad files, and then look up the reputation on each at time T. We could then determine our true positive, true negative, false positive, and false negative rates on these files at time T. We call this approach a “cumulative” approach because it takes into accounts all of the cumulative knowledge we have about each file at the time of classification.

There biggest drawback of the Cumulative Approach is that it doesn’t measure the true protection and false positive impact of the system on the actual user base. Why? Imagine that at time T, we query our reputation system about two good files, A.exe (which has one million users) and B.exe (which has 5 users), and that our reputation system had a false positive on the first file, A.exe, but classifies the second file, B.exe correctly. Based on just these two files (an admittedly small sample set), our cumulative measurement would give us a false positive rate of 50%, since we classified one file correctly and one file incorrectly; in the cumulative measurement approach, both false positives are given equal weight. However, in reality, the impact of a false positive on file A.exe is actually 200,000 times more significant for our customers than a false positive on B.exe!

Thus, this approach provides a population independent assessment of how many unique files can be classified correctly, but doesn’t necessarily reflect the impact of such a false positive (or a true positive) on a population of real users. The other shortcoming of the cumulative approach is that it ignores the fact that we expect a file’s reputation to change over time. As we gather more data about a file, we update its reputation score. A file that has a poor reputation score today might have very good reputation score in two weeks.

Our Interactive-based measurement approach attempts to measure the correctness of our reputation system’s classifications at a per transaction level rather than at a per file level. In this measurement approach, we take a random sample of thousands of queries to our reputation system   many of these queries may actually be queries about the same file, since different users often use the same files over time. We then compute how many of these queries resulted in correct classifications.

So, imagine that files G1, G2, G3, G1, B1, G2, G1, and G1 were downloaded by various users over the course of the study, with G indicating good files, and B indicating bad files. All 4 G1s are exactly the same file, just queried at different times of the day by different people. Each time a user queries about a file, our reputation cloud returns a reputation rating which can be either correct or incorrect. Each incorrect result is obviously a bad thing.

Table 1

Interactive-based measurement approach

Time	7am	8am	9am	10am	11am	12pm	1pm	2pm
File	G1.EXE	G2.EXE	G3.EXE	G1.EXE	B1.EXE	G2.EXE	G1.EXE	G1.EXE
Reputation rating at the time of query	good	good	good	bad	bad	good	bad	bad
Correct?	Yes	Yes	Yes	No	Yes	No	No	No

Now let's imagine that our reputation system returned the following classifications for these eight downloads: good, good, good, bad, bad, good, bad, bad. (Notice how our reputation system actually called G1 good the first time, then called it bad on the last three queries; this is possible, since reputation ratings are constantly evolving over time based on the usage patterns of our users.)

In this hypothetical example, our overall correctness measured using the "Interactive" technique would be 5/8 or 63%. Our true positive rate would be 100% (since we properly classified the one bad file correctly). Our false positive rate would be 3/8 or 38% since we got 3 of our 8 classifications wrong (on the last 3 G1s). Notice that all of our false positives in this example were on file G1. So, while the Cumulative measurement scheme would call this a single false positive, yielding a false positive rate of 33% (the only misclassified file was G1 out of the three distinct files G1, G2, and G3), our Interactive measurement approach would call this a 38% false positive rate since 3 of 8 of our querying users were impacted by this false positive.

We argue that the Interactive approach is more reflective of how well our system is doing in the real world since it measures what percentage of users will actually be impacted by a false positive (or true positive, or false negative) at the time their computer queries about a file. Repeated incorrect classifications are much worse than a single misclassification on a file used by few users. Not only do we measure the system's efficacy in this manner, but our reputation scoring algorithms are designed to minimize the expected interactive error.

New types of positives: "Quasi False Positive" and "Quasi True Positive"

Since our reputation based system assigns ratings to all executable files, both good and bad, any measurement of effectiveness cannot just focus on detection rates of malware. Instead, our analysis must measure the system's overall correctness across all files, both good and bad.

Internally, our system assigns each program a score of between -128 to +128, with more negative numbers indicating a higher likelihood that a file is malicious and more positive numbers indicating a higher likelihood that a file is good; scores closer to zero indicate that our system is less confident in its classification.

With respect to our Download Insight feature, each product has two thresholds that are used to determine which downloaded files are considered "bad," which files are considered "unproven," and which files are considered "good." For example, an installation of SEP might have thresholds of -20 and +25. These thresholds would specify that files with a score of <= -20 are considered bad, files between -19 and +24 are considered unproven, and files with a score of >= 25 would be considered good. (Administrators can also specify policies based on prevalence and age, but these will not be considered in this section since these policies don't technically have a true positive or false positive rate; they simply enable an arbitrary blocking or prompting policy based on the administrator's criteria).

Once the administrator has defined these two thresholds (e.g., -20, +25), they may then assign blocking/prompting policies for files in each of these three regions; for example, they could choose to block all "bad" software, prompt the user for all "unproven" software, and log (but allow) the introduction of all "good" software. In another enterprise, the administrator might choose to block all "bad" and "unproven" software, and log all good software downloads. Therefore, there are three different actions that can take place on a file:

1. The product can outright block a downloaded file
2. The product can prompt the user (providing reputation, prevalence and age details about the file) and allow the user to decide whether or not to proceed with a download
3. The product can allow unfettered use of the downloaded file

Based on the above actions and the user's policy settings, there are 6 potential classification outcomes:

Outcomes on good download:

1. If our system correctly allows a good file to run without blocking or prompting, this is considered a true negative.
2. If our system incorrectly blocks a good file from running, this is considered a false positive.
3. If our system incorrectly prompts the user about a good file, we call this a "quasi false positive," since technically, we have not blocked the good file, but we have inconvenienced the user and indicated that the file is unproven (and likely bad).

Outcomes on bad download:

1. If our system correctly blocks a bad file, this is considered a true positive (a detection).
2. If our system incorrectly allows a bad program to run without blocking it or prompting the user, this is considered a false negative (a miss).
3. If our system prompts the user about a bad file, we call this a "quasi true positive," since we didn't block the file outright, but we did warn the user about its safety via a prompt.

Reputation VIDs (or lack thereof!)

Reputation based detections are inferential in nature (i.e., they are used to determine the presence of a threat without actually identifying that threat by name). As a result, we have had a number of customers get confused when we do a conviction based entirely on reputation since we are not identifying a particular threat by its actual characteristics, but rather by the track record of the application across our entire user base. This type of confusion is not entirely new to the anti malware industry. For example, behavioral detection and heuristic technologies have the same limitations, though arguably less so (especially since it is still possible to identify behaviors or static heuristics that are unique to a particular threat family). Within the context of reputation, it is entirely possible to say that an application poses a threat without any information on the threat it poses. It is not immediately clear how to deal with this type of challenge. While we can ultimately take samples blocked by reputation and analyze them further, we may not have this information at the moment of impact.

Discrepancies and user confusion

Since we have multiple engines in our protection stack, each of which operates at a different point (and with different degrees of information), it is possible to send ostensibly conflicting messages to a customer. For example, on download, we may not have enough information to call a file bad based purely on its reputation (the file might not have an obviously bad reputation, but its reputation might not be sterling either). Without a compelling reason to think the file is bad, we might let it through. Once the file starts executing, however, our behavioral engines will start to observe its actions. Suppose the file now exhibits suspicious behaviors, and that this information coupled with the file's lack of impeccable reputation characteristics are enough to call it malicious. Doing so will potentially confuse the user since we first let the file through on download, only to convict it possibly moments later. From a pure technical perspective, this discrepancy makes sense since a behavioral engine gets to observe more context (namely the application's actions on the system) beyond what we would see on the initial download. As we gather more information about an application, our inclination about whether it is good or bad (as well as our confidence in that belief) will also change. On the flip side, this distinction might be too subtle for a typical customer to make sense of. On the one hand, we could make adjustments to how we score applications so that discrepancies are rare — for example, by resorting to greatest common denominator. On the other hand, adjusting scores in this manner might hurt efficacy. We are still grappling with this issue.

Reputation and maliciousness do not always coincide

A file's reputation score is really a measure of whether its track record (when viewed using telemetry data from a large user base) is consistent with that of legitimate applications. For the most part, a file's reputation score will coincide with whether or not that file represents a threat to the end user (e.g., many threats are extensively polymorphic, causing them to have no real track record whereas many legitimate applications are found on numerous machines). There are, however, circumstances in which these two notions will differ. For example, a brand new application might initially have a low reputation score.

In many enterprise situations, this distinction is less material. In particular, a typical IT Security manager might be OK with blocking access to any applications that have not built up an appropriate track record (so long as all the bad applications are blocked). While blocking some lesser known, but still legitimate, applications will be a byproduct of this policy, the tradeoffs may be well worth it given that malware (especially lesser known targeted malware) can be blocked as well. At the same time, this type of security policy is of little solace to a small software vendor whose applications are blocked. We have tried to deal with this issue through a few mechanisms. First, an enterprise will have the ability to whitelist specific applications and hosting software domains. Second, we calculate reputation scores using a variety of metrics, beyond just a file's prevalence (in fact, in many instances, by leveraging other characteristics, we can call a file good even if it is on only a handful of machines). Third, we proactively look for smaller vendors and automatically download/analyze their applications. Finally, we have a vendor dispute system where applications that we have incorrectly blocked can be submitted.

Effectiveness Results

In this section, we present the results of various effectiveness studies conducted since the release of reputation technology back in September 2009. First, we look at why the features that are used in the reputation system are useful in detecting malware. We then present the effectiveness results from in field protection perspective.

Reputation feature results

As explained in earlier sections, reputation system uses a variety of file attributes to classify a file. We analyzed the reputation Web service logs to over different time periods to derive statistics presented in this section. This section looks at three example features and presents statistics on how they might help detect malware.

Prevalence

The main driver for building a reputation based system was to detect polymorphic malware. While any feature alone cannot be used a sole metric to conclude maliciousness, we observed that the median prevalence for bad files amongst the Symantec user base is less than 50. In contrast, the median prevalence for good files is in hundreds (or thousands) or users. As another data point, over 33% percent of malware strains (as identified by its SHA2 hash) detected during a one day period had less than 5 users!

Origin

A file's origin (where it was downloaded from) is considered as one of the features in the reputation system. After analyzing 128 million downloads of files from over 350,000 URLs, we are able to associate very high trust with over 80,000 websites. On a daily basis, these 80,000 URLs account for more than 20% of the queries for downloaded files to the reputation service.

Signing

Other file attributes like the Subject and Issuer fields of the code signing certificate are also used as features in the reputation system. Just as we can associate reputation with a source website, it also possible to associate reputations with signer and issuer of a code certificate depending on the track record of the signer and issuer. For a one day period, we received 416 million queries that had signer information and found that 415 million of these were almost certainly good based on our analyses. These 416 million queries represented 2.88 million distinct files (as identified by their full file SHA2); 2.85 million of these unique files were found to be good. This indicates that a high proportion of signed software is legitimate (even though we have isolated cases where code signing is used on malware). Hence, digital signatures are a useful feature in a reputation system.

In-field protection results

This section discusses overall in field detection rate observed for various use cases for the reputation engine mentioned earlier.

Download Insight

As of July 1st, 2010, download insight has blocked a total of **6.7 million** threats with daily average (over the last 30 days) of **39,000**. By design, this means that traditional scanning engines on the client are missing this many threats/day, since the reputation system is queried after the scanning engines have analyzed each file. This is first real world corroboration of the magnitude of polymorphic malware as well as the effectiveness of a reputation-based system to address these threats. The overall system efficacy is listed in table 2. It is important to point out that these detections are complementary and additive to all of Symantec's other protection technologies.

Table 2

Download Insight

	Good	Unknown	Bad
Ground Truth: Good	(TN) 97.83%	(QFP) 1.02%	(FP) 1.15%
Ground Truth: Bad	(FN) 22.14%	(QTP) 6.22%	(TP) 71.65%

To summarize table 2, we are checking the reputation of every downloaded file before it can be introduced onto a machine and launched, potentially resulting in an infection. Our protection is particularly aggressive in this download use case since the repercussions of blocking a good file on download (i.e., experiencing a false positive) are much less than the typical repercussions of a false positive (e.g., taking out a key system file and causing the machine to crash). The worst result of a false positive on download is that a user may have to manually un-quarantine a file before using it. Moreover, we have found that blocking on ingress (i.e., at the time of introduction) is a particularly effective way to block new malware from infecting a system. Blocking a single malware file on introduction often prevents 10s of additional malware files from being downloaded post infection. These results (above) show that during our study, we were able to block roughly 72% of all malware downloads, solely using reputation, while only blocking 1/100 legitimate downloads and warning on an additional 1/100 legitimate downloads. And again, this 72% blocking rate is above and beyond the protection afforded by our traditional antivirus engines. This is malware that we believe the entire industry is missing with fingerprints and heuristics.

Heuristics and behavioral corroboration

As mentioned in earlier sections, our heuristic and behavioral detection engines leverage reputation data to improve its accuracy. As of June 2010, we observed that they blocked approximately **50,000** threats per day.

Conclusion

In conclusion, we have noticed that the reputation based system is extremely effective in detecting polymorphic malware on download and regularly prevents tens of thousands of infections per day. In addition, we see vastly improved detection rates and lowered FP rates for our other heuristic and behavioral detection techniques that use the reputation based system for corroboration. We also discovered various challenges and pitfalls in measurement of effectiveness of the reputation system and had to new invent techniques to solve those challenges.

Any technical information that is made available by Symantec Corporation is the copyrighted work of Symantec Corporation and is owned by Symantec Corporation.

NO WARRANTY . The technical information is being delivered to you as is and Symantec Corporation makes no warranty as to its accuracy or use. Any use of the technical documentation or the information contained herein is at the risk of the user. Documentation may include technical or other inaccuracies or typographical errors. Symantec reserves the right to make changes without prior notice.

About the authors

Zulfikar Ramzan is formerly a Technical Director, Vijay Seshadri is a Distinguished Engineer, and Carey Nachenberg is a Vice President within Symantec Security Response.

About Symantec

Symantec is a global leader in providing security, storage and systems management solutions to help businesses and consumers secure and manage their information. Headquartered in Cupertino, Calif., Symantec has operations in more than 40 countries. More information is available at www.symantec.com.

For specific country offices and contact numbers, please visit our Web site. For product information in the U.S., call toll-free 1 (800) 745 6054.

Symantec Corporation
World Headquarters
20330 Stevens Creek Blvd.
Cupertino, CA 95014 USA
+1 (408) 517 8000
1 (800) 721 3934
www.symantec.com