Measuring Drive-by Download Defense in Depth[*]

Nathaniel Boggs, Senyao Du, Salvatore J. Stolfo

Columbia University, New York, NY
{boggs, du, sal@cs.columbia.edu

Abstract. Defense in depth is vital as no single security product detects all of today's attacks. To design defense in depth organizations rely on best practices and isolated product reviews with no way to determine the marginal benefit of additional security products. We propose empirically testing security products' detection rates by linking multiple pieces of data such as network traffic, executable files, and an email to the attack that generated all the data. This allows us to directly compare diverse security products and to compute the increase in total detection rate gained by adding a security product to a defense in depth strategy not just its stand alone detection rate. This approach provides an automated means of evaluating risks and the security posture of alternative security architectures. We perform an experiment implementing this approach for real drive-by download attacks found in a real time email spam feed and compare over 40 security products and human click-through rates by linking email, URL, network content, and executable file attack data.

Keywords: Metrics, Defense in Depth, Drive-by Download, Measuring Security

1 Introduction

The modern Chief Security Officer's (CSO) primary goal to secure an organization in a cost effective manner is frustrated by a lack of formal empirical data suitable for optimizing defense in depth architecture. Purchase price, maintenance costs, cost of false positives on productivity, and the costs of damages prevented by attacks blocked should all be taken into account during purchases of security products. While all these aspects of the cost of security products are important, in this work we focus on measurement of attacks blocked by various products. CSOs can follow expert knowledge codified in best practices and compliance standards such as HIPAA and PCI. This leads to deployment of many

[*] This work is sponsored in part by Air Force Office of Scientific Research (AFOSR) grant FA9550-12-1-0162 "Designing for Measurable Security." The views and conclusions contained herein are those of the authors and should not be interpreted as necessarily representing the official policies or endorsements, either expressed or implied, of AFOSR.

security products such as firewalls, intrusion detection systems (IDS), antivirus, web application firewalls, patch management, and others to provide defense in depth where a particular attack must evade many security products in order to be successful. Intuitively the hope is that attacks that one security product might miss will be detected by another, but such intuitive best practice assumptions often fail or are incomplete as seen in a recent quantitative study of password policy assumptions [1]

To illustrate the current ad hoc nature of defense in depth deployment, consider the following hypothetical scenarios depicted in Figure 1. Imagine an organization with security products deployed across three layers of defense: a firewall, an IDS/IPS, and host antivirus. Assume this organization suffers many attacks against its hosts which could be detected by any of these three layers. In these figures, the red squares represent attacks. For each security product consider it capable of detecting any attacks that fall within its circle. Consider two scenarios presented notionally in Figures 1A and 1B. Figure 1A illustrates the intuitive hope of defense in depth where each security product is able to detect additional attacks and few attacks are able to bypass all three. Figure 1B shows a pessimistic view where the security products only detect the same group of easily detected attacks while leaving the organization vulnerable.

Additionally, consider the scenarios in Figures 1A, 1C and 1D. In these figures, we have not only different layers of defense but consider specific products. Figure 1A represents an ideal setup, but consider an organization whose current products are as seen in Figure 1C. Once aware of the gap in coverage seen here, the organization could find a product that closes their specific gap. This is seen in Figure 1D where the organization may replace firewall 3 with firewall 2 to achieve better coverage. While similar in terms of detection rate and absolute number of attacks detected, firewall 2 and firewall 3 differ in terms of which attacks they block. Current product testing only shows per product detection rates without indication of what overlap might exist with other products.

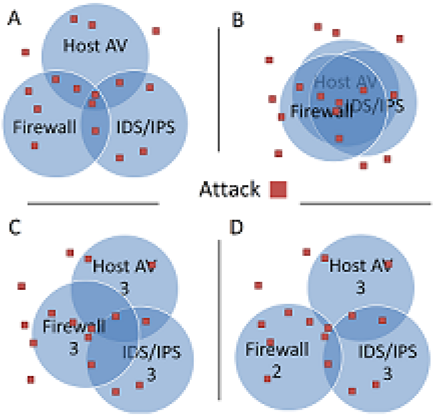

Fig. 1: Illustration of overlap and total coverage of four defense in depth deployments. Each square represents an attack. Each circle is a security product. Attacks in a circle mean that those attacks will be detected by that security product.

We address this issue by expanding the standard security product detection rate testing in two ways. First, we track not only individual security product detection rates, but also which attacks each security product detects.

This allows us to take a union of the sets of attacks an arbitrary group of security products detect in order to calculate the coverage or total detection rate a set of security products have in aggregate against a set of attacks. Second, rather than testing security products against isolated pieces of attack data, we carefully record and link all data from a particular attack in its various forms so that security products from multiple layers of defense can be tested against the same attack.

This new approach to data gathering and testing could both validate the need for a new security product and give a baseline to quantitatively measure how much that security product improves the overall security posture of the organization by calculating the increase in coverage (total detection rate) after adding that product. Any inexpensive security product that is complementary to the organizations' current defense in depth becomes a good investment while expensive products that add little relative improvement could be used only to protect the most valuable of assets. Additionally, since the data sets and security products only have to be gathered and tested by one security service provider regardless of how many organizations make use of the data, this approach is scalable with costs amortized across any number of benefiting organizations.

In order to demonstrate the useful knowledge that is possible to gain from our methodology, we created a prototype system, we call Security Posture Integration and Correlation Engine (SPICE). SPICE covers four layers of security tested in near real time (within a few minutes in most cases) against real in the wild drive-by download attacks originating in widespread spam emails. To prevent biasing results by using existing known malware, we use a real time spam feed from Abusix [2] to send links to an instance of the Cuckoo Sandbox [3] honeypot driving full virtual machines designed to be vulnerable to common in the wild drive-by download exploits. The files, URLs visited, and full network packet capture are then logged and linked to the email that sent the malicious URL. We integrate a fifth layer of security, human click-through rates, into the system via a user study utilizing the same attack data allowing us to directly link the believability of spam emails to detection of the attack at other layers. While this experiment and its data collection is specific to the widespread drive-by download attack vector, we discuss applying our methodology (linking attack data across layers and tracking individual attack detection by security products rather than aggregate detection rates) to other attackers and attack vectors in Section 2. The key innovation in SPICE over existing malware collection is that we systematically and automatically link the data captured. Knowing for example which HTTP link led to which Windows PE file being loaded onto a victim machine allows us to compare a domain reputation system such as Google SafeWeb to a host or network antivirus product. We can determine that the attack would have been blocked had either successfully identified it.

This paper provides the following contributions:

1. We describe a practical methodology for calculating the coverage (total detection rate) of a group of security products deployed across many layers of

security as defense in depth as well as the marginal detection rate gained by adding a new security product.
2. We apply this methodology to create SPICE, a prototype system, collecting drive-by download attacks in near real time linking emails, domain names, network traffic, and executables to each attack.
3. With this data, we test over 40 security products across many layers of defense. We report individual detection rates and delayed detections as well as the correlation and total coverage of sets of security products.
4. Using this methodology and prototype we are able to compare the results of a user study on human click through rates for actual attack emails to other security layers.
5. The data we have gathered will be provided to the research community. The linked attack components and detailed detection results may offer new research opportunities for the research community.

The remainder of this paper is organized as follows. Section 2 discusses definitions, overall approach, and the challenges facing such comprehensive measurement. Section 3 describes the SPICE system architecture. Section 4 presents detection rate, coverage, correlation results, and user study results. Section 5 discusses related work on defense in depth, drive-by downloads, measurement experiments, and best practices. Section 6 mentions future goals and directions going forward. Finally, section 7 summarizes the findings and approach.

2 Methodology

2.1 Approach

The approach for measuring defense in depth consists of a series of steps.

1. Choose a particular attack vector. This paper centers on a prototype experiment for drive-by downloads. Other potential attack vectors include web server attacks, insider attacks, data exfiltration, and so forth.
2. Find security products that are capable of detecting an attack for the chosen attack vector.
3. Set up a honeypot capable of recording all the data from the attack that each of these security products take as input. For example, ensure the honeypot can capture the network traffic to test a network IDS and executable files to test antivirus or host detectors.
4. Collect attack data and normal data if possible. In the case of testing against widespread attacks as we do with SPICE this could come from a commercial spam feed or existing honeypot. Real time in the wild data is ideal for accurately measuring existing attacks.
5. Have the honeypot record all the attacks in real time and monitor for successful attacks. Save all the data related to successful attacks and link the attack components.

6. Test each security product against appropriate attack components of the data collected and record the results. Use these results to determine which security products detected which attacks.
7. With a set of detected attacks for each security product one can answer questions such as what is the coverage (total detection rate) of a group of security products, the marginal increase in detection rate by adding each security product, and which ones detect the most difficult attacks.

This process is repeatable for each attack vector to develop a broad view of the security posture of the defenses although additional attack vector specific data collection methods would be required. The most difficult task is data collection, which in turns depends on the type of attacker an organization is interested in defending against. SPICE is as reliant as any other testing framework on good underlying data and ground truth.

2.2 Definitions

Attack: An instance of an attacker attempting to gain unauthorized access to a system. For this experiment, we more specifically define an attack as a single instance of a virtual machine in a honeypot visiting a drive-by download website and getting infected with an executable file.

Attack Cluster: A group of similar attacks presumably launched by the same attacker. For SPICE, we group attacks into attack clusters based on email contents[1].

Layer: In this work we use 'layer' to refer to all the security products deployed in a defense in depth architecture that use a particular type of data. For instance, a defense in depth architecture with regards to the drive-by download attack vector is made up of many layers such as security products operating on email content, security products operating on network traffic, etc.

2.3 Linking Attack Vector Data

We link attack data so that security products from different layers can be tested against the same attack based on whatever data that attack generates suitable for each layer. Recording and linking this data takes different forms based on the attack vector. For instance in the drive-by download scenario one would capture the spam email, the initial malicious link, the network traffic as a virtual machine visits the link and gets infected, and the files and processes loaded onto the victim machine. For web application attacks, one would capture incoming network packets, reassembled HTTP requests, server host data such as file

[1] We cluster emails with a similarity score greater than .8 computed between two emails' content by taking the ratio of the sum of the lengths minus the Levenshtein distance with weight two for character replacement and the sum of the lengths.

system accesses and system calls, network traffic to the database backend, and database queries linking all this data together by time window and other signatures such as IP address and process ID. For the insider threat attack vector, we could track user activity logging into each server, file transfers, host data such as file access and system calls, and outbound network traffic linking this data together by user login and machine IP address. Each security product is tested as to whether it can detect the attack through whichever piece(s) of this data it is designed to process. For example, user education would be tested on the spam email content from the attack while a network intrusion prevention system would read the network data from the same attack.

While SPICE deals only with the initial attack vector data crossing four layers of security, additional layers of security play an important role, especially against more sophisticated adversaries. The stages of an attack after initial infection such as propagation, downloading additional malicious features, and eventually data capture and exfiltration provide the potential for many more layers of security to accurately detect some stage of an attack. SPICE can incorporate these additional layers of data if additional data collection capabilities are added. The most straightforward extension would be simply to leave any infected honeypots online and observe future behaviors. This could capture any generic botnet traffic, but without exposing any sensitive data. We believe that a more sophisticated honeypot solution where attacks are run on servers with enticing but fake data such as described in [4] could be constructed. This would allow for excellent data collection and the ability to accurately test security products designed for data loss prevention or multistage infections.

2.4 Discussion of Data Sets and Adversarial Capabilities

We define each class of adversary by the additional resources/capabilities they bring to bear. While certainly not perfect, we believe that definitions along these lines provide clear means of separation and data gathering while still giving organizations a good idea of the level of sophistication of adversaries they are vulnerable to. Some of these adversaries lend themselves to easy data collection using honeypots collecting real data on the internet. Others, especially as they grow in sophistication raise the cost of in the wild collection with sparser data and the expense of fielding sophisticated honeypots. For many of these we propose approximate synthetic data set generation that while not ideal could at least provide a picture of the threat. Note that the scope of this experiment is limited to the widespread untargeted exploit kit attackers as described below.

Widespread Untargeted Exploit Kit Attackers One drive-by download adversary class is what we term a widespread untargeted exploit kit attacker. This is an adversary that not only makes general nontargeted attacks, but does so against any email address available. A perfectly representative data set is then easily collected from honeypot emails. The only challenge in modeling such an adversary is sorting through the enourmous amounts of non-harmful spam to find those spam emails that contain malicious links. These links seem to point to standard (and perhaps even outdated) exploit kits with well known exploits and mostly not so subtle executable payloads. What makes this adversary dangerous is that an organization must be prepared to deal with the attacks as they are widespread enough that they will certainly reach the organization. This is the adversary class that our experimental data set represents.

Targeted Exploit Kit Attackers Another class of adversary is one we term as targeted exploit kit user. This adversary would still lack the innovation and/or resources to field any true zero-day exploits, but is capable of using the available attack tools to their fullest. This adversary also is capable of making targeted attacks using spear-phishing and brand new clean domains in order to compromise higher value targets. This is the level of adversary where inexpensive security products like domain reputation and email spam filtering start to fail. Classes of security products such as fully patched systems, sophisticated antivirus with whitelisting and behavioral analysis, host IDS, honey files, and data loss prevention systems become important if not necessary. Modeling this adversary class with in-the-wild data becomes difficult unless an organization has the opportunities to capture such attacks with its own honeypots for later analysis. Due to this class of adversary's reliance on existing exploit kits and available tools, we believe that a reasonable data set could be manufactured using those same tools. The challenge here is to maintain up to date copies of widely available attack tools and exploit kits especially when new exploits are added to the exploit kits before patches are widely available.

Zero-day, insiders, and more sophisticated attackers As attackers increase in sophisticated data goes from hard to collect to nearly impossible. Even if any such data becomes available, it would likely be of a historical nature and scarce. Even historical data could be interesting to test with as security products could have their updates rolled back to a date before the data was sent. Any conclusions drawn based on historical data would still be suspect as the nature of sophisticated attacks is to be fairly unique.

3 System Architecture

The Security Posture Integration and Correlation Engine (SPICE) prototype system, depicted in Figure 2 extends a traditional honeypot by carefully linking attack data across different layers in a database such that security products even from different layers of defense can all be compared. In this way, the coverage of an organization's defense in depth architecture can be evaluated. As an attack hits the honeypot, pieces of it are logged and then scanned by the appropriate security products once confirmed as malicious. As these pieces are linked, we can determine which security products detect the attack even if they run on different pieces of data linked to the same attack.

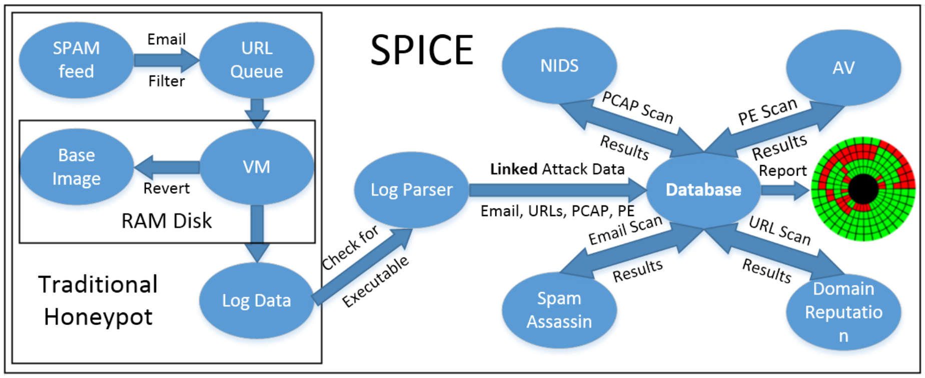

Fig. 2: An overview of SPICE. We build upon a traditional honeypot by carefully linking attack data in order to see how security products from different layers of defense overlap or complement each other.

We use an existing spam stream from spamfeed.me by Abusix [2]. It is a real-time spam feed that is captured through a large number of spam traps online. We receive 1 million emails a day. Each email is individually parsed, and all URLs from that email are then extracted and put into a database. Unfortunately, while such a feed guarantees that the emails are spam, the distribution of spam is highly skewed. Of the emails that even had links, the vast majority point to pharmaceutical spam with only a fraction of a percent serving active malicious content, handfuls per day from unique sites. We filter out links visiting only one from each domain for any twelve hour period in order to reduce the load on the VM clusters visiting each link.

To visit the URLs in emails, we use four clusters of virtual machines (VM), with 40 in each cluster that run on top of VirtualBox [5] across two physical machines. These virtual machines run off RAMdisk to minimize the impact of disk IO towards running and reverting virtual machines. Each cluster of virtual machine has its own configuration, with variations of browsers installed and its plugins such as Java, Adobe Flash, Adobe Acrobat Reader. We then validate each cluster's setup against CANVAS [6] a white hat penetration testing tool, making sure they are indeed vulnerable to existing exploits known to be targeted by exploit kits. We use Cuckoo Sandbox [3] to drive these virtual machines to visit each link logging host activity and new files created.

Each cluster has it own driver, which takes the URL feed and instructs the browser inside each machine to visit the link. We visit each URL three times per VM cluster, to compensate for the instability of the exploits. Sometimes exploits fail to infect even vulnerable machines, most likely due to poor code or nondeterministic exploit conditions. After waiting at least one minute, the VM is reverted to a clean state.

Each time, a VM visits a link the honeypot generates a log that contains a pcap file of the network traffic generated, and records details of the execution of programs in the operating system as well as new files generated. SPICE scans log files to determine if any executable files have been generated. The appearance of a new executable file is a strong indicator of an attack successfully occurring and provides a high confidence of malicious behavior, with a zero false positive rate as far as can be determined. Every successfully created new executable is eventually identified as malicious by at least one of the antivirus products tested. This classification is also used in other recent literature on drive-by download attacks [7]. If a new executable is found, we flag this as an attack and store the executable files as well as the email, pcap file, and the URLs visited during that attack linked to the attack in a central database.

We have installed three host based antivirus software programs, which scan the new files within a minute after the file has been logged and inserted into the central database. The antivirus programs rescan the files every six hours to test if new updates to the antivirus program are capable of detecting the attack. All the antivirus software is configured to update regularly. At the same time, we send the executable files to VirusTotal [8] once every 12 hours.

On top of scanning the executable files, we have the email and pcap file that are associated with the attack. We run Spam Assassin [9] on the email and Snort [10] a network IDS on the pcap file. Both of them follow the same rule for processing the incoming data: within a minute after arrival and rescans at 6 hour intervals. Spam assassin is configured to use the most up to date rules from the Internet automatically. Snort updates its rules daily from the Emerging Threats [11] public rulesets.

We also test domain reputation systems. We use the domain reputation data from four public domain blacklists to test each link that is associated with the attack to see if the domain is flagged as malicious. Whenever there is any result from the scanning system, they are added to the database for further analysis and rescanned periodically.

In order to test the efficacy of security education for users in an organization, we conducted a user study to measure the human factor in drive-by download attacks such as performed in [12]. User education has the potential to be a beneficial layer of defense as it complements almost all existing layers. If users can spot suspicious links and not click on them in the first place then the attack is thwarted. To test the likelihood of users clicking on a malicious link in the spam emails, we took one email from each of the attack clusters. After adding a unique identifier to and changing the malicious link to point to a benign web server with an unaffiliated domain name, we then sent the email to 10 randomly selected users from (anonymized) (IRB approval was received and will be cited) for each cluster resulting in a total of 360 emails sent. We send these emails from a clean account with an old nonmalicious domain for the links to evade reputation based spam filters, but content based spam filters could still affect this experiment so the results will represent more of a lower bound. If a user visited that unique link we recorded the click through and displayed a webpage detailing the study as well as warning of the dangers of clicking unknown links. By using the same attack data used by the other security products for testing, we can use this user study to see how human click-through rates overlap with other attack detection. In the future, one could experiment with different user education techniques and by comparing their effectiveness directly against other security products one could determine whether user education is more cost effective than buying additional technical security products.

In summary, the entire list of products employed in SPICE includes:

1. One email spam detector
2. Human spam click through measurement
3. Four domain reputation systems
4. One network IDS
5. Three host antivirus programs
6. Forty antivirus engines via VirusTotal [8]

4 Results

4.1 Attack Data Collected

The 1463 virtual machines infected in the course of the five and a half week experiment cover attacks from 730 unique emails. With an average of about two infections per email out of the twelve times each link is visited (three per four virtual machine setups) we note that capturing in the wild drive-by download attacks is not reliable. The low rate of infections can be attributed to a combination of factors that make precise measurements difficult. A particular attacker may not have an effective exploit for some versions of vulnerable software. We mitigate this by confirming that in the wild exploits target the versions of vulnerable software each virtual machine setup runs. An attacker may blacklist repeat visits from the same IP or to the same unique link to thwart probing. In this case the best we can hope for is to be infected the first time. Some exploits are more reliable than others. Also, an exploit can fail or take longer to compromise a virtual machine than SPICE monitors.

Associated with these attacks are 942 distinct domains visited by virtual machines during the course of the infection and 576 unique executable files. The overlap of exactly the same files being used by attacks that started with separate emails indicates that many of these attacks are originating from the same attack campaign. In fact, most emails end up being a slight polymorphic variation of each other presumably to evade basic exact match spam filters. Once we cluster these similar emails (see Section 2.2 for details) we derive only 36 clusters. Unlike phishing emails or pharmaceutical spam, these emails' sole purpose is to get a user to visit the URL, which then launches a drive-by download attack.

In reporting successful detection of an attack cluster for a security product, we choose a pessimistic view. As launching additional attacks is inexpensive for an attacker, if a security product fails to raise at least one alert per attack in that cluster, then we claim that that security product does not detect the attack. We

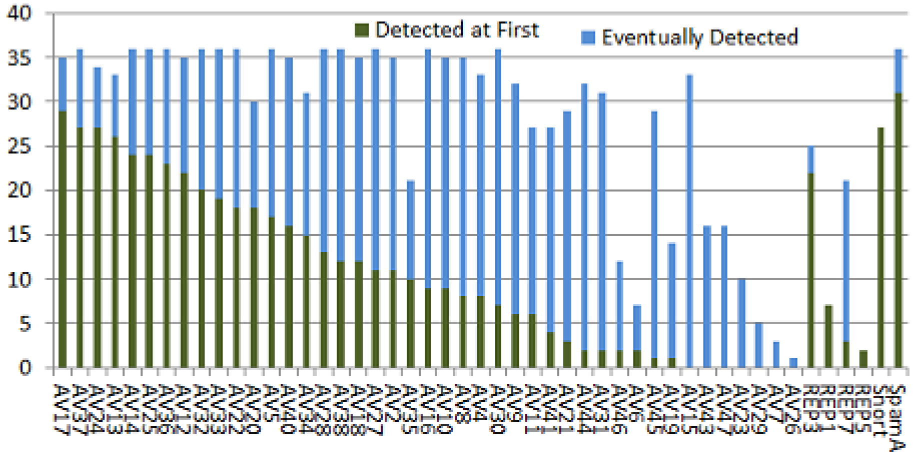

Fig. 3: Cluster detections per security product. Eventually detected attacks are ones that the product did not detect at first scan, but did detect on a later scan.

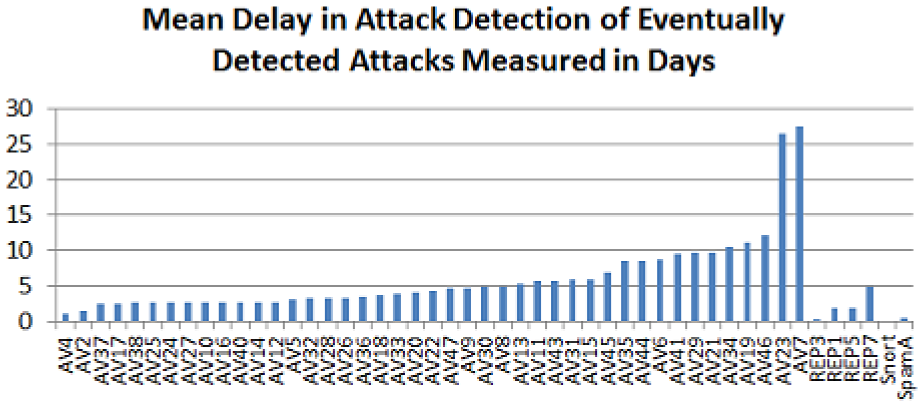

Fig. 4: Average (mean) number of days after an attack occurs that it is detected if it was not detected at first but eventually was.

believe that this is the most realistic scenario since as long as one of the attacks gets through then the adversary succeeds.

Clustering attacks based on email similarity yields a view untainted by repeated high volume attacks seen in Figure 3. We anonymize the commercial products in order to avoid any apparent bias or endorsement. The graph shows the number of attack clusters detected at first scan as well as those eventually detected, which is discussed further below. A wide range of individual security product effectiveness is displayed varying from products completely ineffective upon first scan to those that detect the vast majority of attack clusters. Here we see that while many security products still perform well, a need exists for multiple security products to fill gaps in coverage as no single security product detects all attacks.

4.2 Late Detections

Although the initial detection of an attack is most important and what is typically measured, we also continue to test security products over time to see if they eventually detect attacks that they initially missed. See these eventually detected attacks in Figure 3. For the initial test of each attack's appropriate data (file for antivirus, PCAP for NIDS, etc) against each security product we averaged 11.3 detected of the 36 clusters. In the weeks following the initial infection as we daily retested each attack that average eventually swelled to 27.3 detections of the 36 clusters per security product. The mean time between the attack occurring and the rescan that successfully detected previously undetected attack is 5.7 days on average for the security products we tested. This does vary significantly by security product as seen in Figure 4 with some security products averaging less than a day delay and others taking upwards of 25 days. This striking result confirms the need for testing to be done in real time as a significant delay can radically alter the results. The security products tested via VirusTotal may even have inflated initial detection rates as the hours delayed between capturing a new malicious file and their servers testing it could give vendors enough time

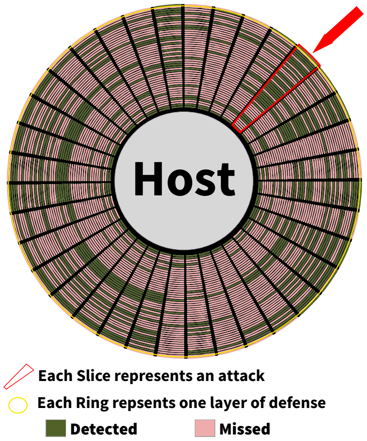

Fig. 5: Graphic display of the first time each security product is tested with an attack cluster. Each concentric circle is a security product and each arc is one attack cluster. If that security product detects the attack cluster then the intersection of the circle and arc is dark otherwise it is light.

to add new signatures that would not have been present during a real time attack. Based on the large number of attacks that are missed initially by security products on VirusTotal but eventually detected, we suspect that most vendors have a significant lag time in updates. While the delayed detection is certainly not ideal, this pattern of eventually detecting attacks could perhaps be leveraged into a system that saves and rescans data for an organization in order to detect what machines have been compromised in the past.

4.3 Correlation of Security Products

One of the key questions we set out to answer is whether and to what extent security products or particular defensive layers overlap i.e. do security products all tend to detect the same easy to detect attacks or do security products tend to detect separate attacks from each other. Another way to put this question is are the detection results of security products independent, or negatively or positively correlated that is do two security products, which each detect 90% of attacks separately, together detect 99%, closer to 100%, or closer to 90% of attacks respectively. By tracking not only the detection rates, but also which attacks each security product detections even across different layers of security by linking the data each attack generates, we have the data to answer these questions for the security products tested here. We present this data in three Figures 5, 6, 7.

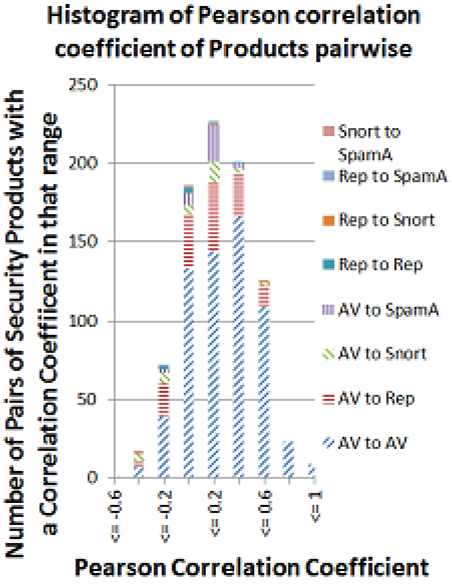

Fig. 6: Histogram of Pearson correlation coefficients of each pair of security products. A higher coefficient means that the two products overlap more.

The first figure illustrating the correlation or overlap of the security products is Figure 5. In this figure, we show data similar to Figure 3 except that instead of each security product being a column, each security product is a concentric circle with its detection results lined up such that each arc of all the circles represents the same attack cluster. In this way rather than showing the detection rate of each security product, we can easily see which security products detect the same or different attacks. By drawing a line from outside the circle to the center, one can see which security products detected the attack represented by that arc. In order to have detected all attacks with at least one security product, one would have to choose a sufficient number of concentric circles such that any line drawn from outside the circle to its center is detected by at least one concentric circle. This visualization has its roots in a common metaphor using a castle to illustrate layers of defense. Consider each security product represented by a concentric circle as a moat or castle wall and many armies (attacks) surrounding it each attacking its own section. A detection means that the particular army (attack) is unable to breach that security product.

Figure 6 uses the Pearson product-moment correlation coefficient to obtain a numerical measurement that represents the linear dependence between the attack cluster detections of a pair of security products. The coefficient is calculated for each pair of security products tested and the results are plotted as a histogram in Figure 6. The coefficient would be 1 in the case that the detection results are identical (completely overlap) and −1 if they were opposite (no overlap). On average the results show that security products tend to overlap slightly more than independent security products would. This means that by adding an average new security product to existing defense in depth we would expect it to increase the total detection rate relative to its absolute individual detection rate, but by less than if it was randomly detecting attacks at its individual detection rate. For example, if an existing security product detected 90% of attacks and one added a security product with Pearson coefficient of 0.2 with regards to the existing security product and a detection rate of 80%, one would expect slightly less than 98% combined detection rate. While skewed slightly towards positive correlations, these results show a number of negative correlations where two security products combined perform better than if their total detection probability was a simple product of their detection rates. For instance in the above example this would mean that the total detection rate would be above 98% instead.

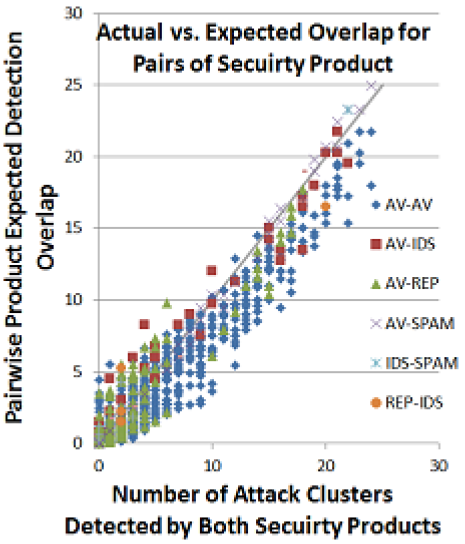

Fig. 7: Plot of the expected value of the overlap between each pair of security products if those products' detection rates were independent versus the actual overlap. Points below the line show more overlap than random.

Figure 7, also focused on the correlation of security products, graphs the number of attack clusters detected by both security products for each pair of security products versus the expected overlap in their attack cluster detection if each security product detected attacks at random based on its detection rate. Points above the line indicate a pair of security products that are more complementary than random while points below the line indicate a pair of products that overlap more than random detections. This result naturally mirrors the previous histogram of Pearson coefficients in indicating that on average security products appear to be slightly more redundant than if detections were random based on overall detection rates.

Intuitively, most products would use the same techniques and signatures making them mostly redundant, but we find that security products are only slightly redundant on average with many doing as well as completely independent detection mechanisms and some performing even better. While security products seem to vary greatly in their detection rates, even the less effective overall seem to occasionally detect an attack that bypasses most other security products. These results may come from a lack of attack intelligence sharing by the security industry, a wider than expected range of effective proprietary algorithms, or the challenge and chance associated with trying to detect increasingly polymorphic malware [13]. The results indicate that perhaps extensive usage of what might intuitively seem to be redundant security products could in fact significantly increase security. While using multiple inline host sensors is impractical, the results suggest that using multiple domain reputation systems and network based antivirus engines could increase the detection rate of the whole defense in depth strategy.

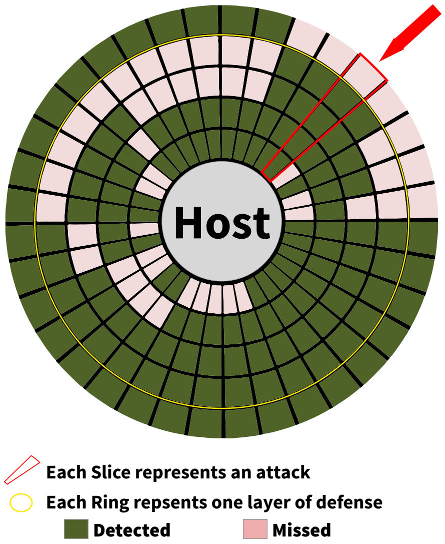

Fig. 8: Graph of top AV (outer ring), top domain reputation system, Snort, Spam Assassin, and human click rate (red if at least 1 in 10 users clicked a link in the email) (inner ring). Each arc is one attack cluster. If that security product detects the attack cluster then the intersection of the circle and arc are dark otherwise it is light.

An advantage of SPICE is that while being able to evaluate the weaknesses of a particular security architecture, the approach also provides direction as to how to mitigate those weaknesses. The ability to find complementary security products is one of the largest contributions of SPICE. An organization rather than blindly picking security products that appear to be good in absolute detection rate, may now determine additional security products that detect attacks that are missed. This simple but crucial shift in evaluating security architecture should help organizations close existing security holes and allocate resources more efficiently. This complementary nature of security products is fundamental to the very idea of defense in depth.

4.4 Human Factor

We ran a user study for all 36 attack clusters sending 10 emails per cluster for a total of 360 emails sent to separate users. See Section 3 for details. After three days we received click throughs on 17 of the 360 unique identifiers sent out. These hits are somewhat focused with 4 of the clusters receiving two click throughs, 20% of the users tested for those clusters. Additional education of those 5% of users who click through if effective could lead to a strong complementary layer of security. Attack clusters with emails that users fall for seem to be roughly as difficult to detect by other means. Compare the inner circle in Figure 8 with security products from other layers. The average attack is detected by about 15 security products while clusters with at least one user clicking on the link are detected by 14 security products on average. This increases slightly if we only take the clusters that at least two users clicked on the link for. From this limited study and widespread spam emails leading to drive-by downloads, we saw no significant correlation in click throughs and other security product detection of attacks. The percentage of users clicking through is lower for the study than other human factor studies [12] perhaps due to the fact that all the spam emails involved here are targeted to a general population rather than targeted to a particular organization or individual. Also note that these results likely represent a lower bound on the number of people who would click on the email as any spam filters in use by the users studied could block the email before it reached them. We took the precautions of sending from a clean source email address and having the links point to a clean domain name in order to mitigate this concern as much as possible, but the email content being that of real attack data could still trigger spam filters.

4.5 Use Case

To further illustrate the usefulness of SPICE consider a hypothetical case study based on the results. Assume a small organization with a CSO who did extensive research and deployed an antivirus and domain reputation system with the best stand-alone detection rates. In the experiment, the best stand alone detection rate for an antivirus was 29/36 attack clusters and for domain reputation 22/36. This is the current state of the art for product comparisons with both of these being best in class at least for this data set. The natural question here is what is the overlap of these two security products and do they together detect all of the attack clusters. SPICE can be used to answer this question. These two security products together managed to detect 33 of the 36 attack clusters (see the outer two circles of Figure 8). Notice that three arcs representing attacks are all light meaning that those three attack clusters went undetected by both security products.

Now assume that this hypothetical CSO wants to improve his organization's security against such widespread attacks. Considering new security products is a proper next step. The network IDS Snort [10] with the Emerging Threat rules [11] detected 27 of these same 36 attack clusters, but more importantly it detects 2 of the 3 attack clusters that were missed by the antivirus and domain reputation system. Measuring the current state of user click throughs we see users did not respond to 23/36 attack clusters including one of the three missed by both the antivirus and domain reputation system. Similarly, we can check Spam Assassin [9] an email spam detection product, which as expected considering the data set is based on widespread spam, had a strong detection rate identifying 31/36 of the attack clusters including all three that the antivirus and domain reputation system failed to detect. See Figure 8 for a visualization of all layers together. In this situation installing Spam Assassin alone covers against these attacks but also one could choose to install Snort for extra redundancy. An organization could only look at sets that already include their existing security products in order to find the next security product to deploy.

5 Related Work

Research on defense in depth often focuses on broad frameworks and the need for defense in depth without providing a specific methodology suitable for experimental measurement and evaluation of defense in depth. General themes such as the need for security products to be at least independent or multiplicative in strength and the need for measurement are repeated in many works [14] [15] [16] [17]. An argument for careful independence assumptions and goals of purposefully choosing security products that are better at detecting different classes of attacks in order to achieve higher coverage than independent security products in presented in [17]. The authors in [14] suggest measuring the attacker's cost to bypass each security product or group of security products. This approach could scale well with regards to sophisticated attackers were real attack data is scarce or outdated making direct measurement of detection rates difficult although methods of measuring such cost are still nebulous. A method for combining the detection rates of security products from all layers is presented in [15] similar to our methodology, but the authors assume each security products independent whereas, we directly measure the overlap between security products without assumptions of independence. While work on directly measuring defense in depth is rare, [18] presents a method for using attack graphs to measure firewalls combined with host vulnerability information to detect holes in a defense in depth deployment.

Drive-by downloads, the attack vector we use in the SPICE prototype system to demonstrate our methodology, is a well studied area. A number of approaches to protecting against drive-by downloads have been presented [7] [19] [20]. We use the same ground truth definition as one recent work [7] as well as its baseline comparison with VirusTotal, which we use. In future work, when we test against more sophisticated attacker data, all of these novel research approaches can be integrated into the SPICE framework as additional layers to test. Studies of using multiple antivirus security products as defense in depth have been conducted such as [21] [22] all showing benefit from combining multiple antivirus engines. These studies are limited to only one layer of defense in depth where with SPICE we expand beyond just antivirus to analyze security products across different layers. Human spam message click through has been studied before such as in [12]. We use similar mechanics with the addition of being able to link the results to other pieces of the same attack that generated the email clicked on.

Some of the most closely related work to SPICE is conducted by commercial security product testers. For example, NSS Labs [23] conducts extensive tests of security products ranging from home user anti-malware solutions to the newest corporate all-inclusive network security appliances. SPICE expands on these existing approaches. By linking the data across layers of an attack vector, we can reason about how products which operate at different layers i.e. network and host detectors complement each other. We have one set of attacks that we test all layers against at once.

The Anti-Malware Testing Standards Organization (AMTSO) [24] creates and maintains best practices for testing security products. While acknowledging the impossibility of a perfect test and evaluation, AMTSO provides guidelines and suggestions for achieving the most accurate results. We implement as much of their advice as possible including one of the most important: real time testing. In the future, we hope to implement some of their additional best practices such as the important false positive measure and running tests on bare metal machines rather than the virtual test environment we use.

Recent extensive efforts to make data more available to the research community such as Symantec's Worldwide Intelligence Network Environment (WINE) [25] and others are crucial to the repeatability of experiments and gaining of new insights into attacks. The ability to access large numbers of samples and meta data will hopefully fuel the next generation of detection algorithms. Unfortunately, such archival data sets do not lend themselves to evaluating current security products. As we see in the results, security products are much better at detecting known threats, but these threats often slip by undetected the first time seen. The data set, which we are releasing to researchers as well while suffering from the same issue of outdated samples as WINE, has one important advantage. We keep track of which samples belong to the same attack chain. Hopefully this additional metadata will help other researchers or be useful in conjunction with larger data sets such as WINE.

6 Future Work

In future work, we are actively adding additional attack vectors starting with web application attacks as discussed in Section 2.3. Also, we wish to measure how security products change over time for all attack vectors. In particular we want to see if security products that are correlated in this data set stay correlated in the first or if such correlations occur by chance or for limited time periods. We want to add software updates as a layer of security by performing studies on vulnerability life times, time to patch, and zero-day attack prevalent. With these parameters an organization could combine their patching practices with their available security products to form a set of security controls that better represents their defense in depth posture. We wish to add cost information such as false positive rates and price to security products so that sets can display total cost in addition to total detection rate. We also would like to use SPICE to test and compare how novel full class prevention security products such as BLADE [7] or virtualization layers might perform. If these security products perform up to their full potential of shutting down whole attack vectors, they may justify their high cost of user training/deployment effort. SPICE could help show how this solutions might succeed compared to multiple commercial solutions that even together suffer many weaknesses.

7 Conclusion

We presented SPICE, a novel method and framework, to measure how secure an organization is by testing real security products with real attacks. By designing additional experiments measuring all known attack vectors and security products an organization uses, we can measure how secure that organization really is. To compute this for a single organization is perhaps prohibitively expensive considering the costs associated with procuring the appropriate attacks to represent sophisticated adversaries and testing infrastructure needed to adequately test advanced security products. Fortunately, the cost can be amortized by a security services provider across a number of organizations that could benefit from the same knowledge of how their existing products complement each other and what new products could fill specific weaknesses they may have. Being able to cost effectively compare security products is crucial. Current tests give no good indication of whether a security product detects the same attacks already detected by existing products especially ones from different layers. SPICE directly measures the underlying assumption of defense in depth that security products complement each other in detecting different attacks. We provide a feasible empirical measurement of an organization's security while at the same time providing the information of which security products would most enhance that organization's security posture.

References

1. M. Weir, S. Aggarwal, M. Collins, and H. Stern, "Testing metrics for password creation policies by attacking large sets of revealed passwords," in *ACM Conference on Computer and Communications Security*, E. Al-Shaer, A. D. Keromytis, and V. Shmatikov, Eds. ACM, 2010, pp. 162–175. [Online]. Available: http://doi.acm.org/10.1145/1866307.1866327

2. http://abusix.org.
3. http://www.cuckoosandbox.org.
4. B. M. Bowen, S. Hershkop, A. D. Keromytis, and S. J. Stolfo, "Baiting inside attackers using decoy documents," in *SecureComm*, ser. Lecture Notes of the Institute for Computer Sciences, Social Informatics and Telecommunications Engineering, Y. Chen, T. Dimitriou, and J. Zhou, Eds., vol. 19. Springer, 2009, pp. 51–70. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-05284-2
5. J. Watson, "Virtualbox: bits and bytes masquerading as machines," *Linux J.*, vol. 2008, no. 166, p. 1, 2008.
6. http://immunityinc.com/products-canvas.shtml.
7. L. Lu, V. Yegneswaran, P. A. Porras, and W. Lee, "BLADE: an attack-agnostic approach for preventing drive-by malware infections," in *ACM Conference on Computer and Communications Security*, E. Al-Shaer, A. D. Keromytis, and V. Shmatikov, Eds. ACM, 2010, pp. 440–450. [Online]. Available: http://doi.acm.org/10.1145/1866307.1866356
8. https://www.virustotal.com/.
9. http://spamassassin.apache.org/.
10. M. Roesch, "Snort, intrusion detection system," *http://www.snort.org*.
11. http://www.emergingthreats.net.
12. B. Bowen, R. Devarajan, and S. Stolfo, "Measuring the human factor of cyber security," in *Technologies for Homeland Security (HST), 2011 IEEE International Conference on*, nov. 2011, pp. 230 –235.
13. Y. Song, M. E. Locasto, A. Stavrou, A. D. Keromytis, and S. J. Stolfo, "On the infeasibility of modeling polymorphic shellcode," in *Proceedings of the 14th ACM conference on Computer and communications security*, ser. CCS '07. New York, NY, USA: ACM, 2007, pp. 541–551. [Online]. Available: http://doi.acm.org/10.1145/1315245.1315312
14. S. Stolfo, S. Bellovin, and D. Evans, "Measuring security," *Security Privacy, IEEE*, vol. 9, no. 3, pp. 60–65, 2011.
15. H. Cavusoglu, B. Mishra, and S. Raghunathan, "A model for evaluating it security investments," *Commun. ACM*, vol. 47, no. 7, pp. 87–92, Jul. 2004. [Online]. Available: http://doi.acm.org/10.1145/1005817.1005828
16. M. Stytz, "Considering defense in depth for software applications," *Security Privacy, IEEE*, vol. 2, no. 1, pp. 72–75, 2004.
17. B. Littlewood and L. Strigini, "Redundancy and diversity in security," in *Computer Security ESORICS 2004*, ser. Lecture Notes in Computer Science, P. Samarati, P. Ryan, D. Gollmann, and R. Molva, Eds. Springer Berlin Heidelberg, 2004, vol. 3193, pp. 423–438. [Online]. Available: http://dx.doi.org/10.1007/978-3-540-30108-0_26
18. R. Lippmann, K. Ingols, C. Scott, K. Piwowarski, K. Kratkiewicz, M. Artz, and R. Cunningham, "Validating and restoring defense in depth using attack graphs," in *Military Communications Conference, 2006. MILCOM 2006. IEEE*, 2006, pp. 1–10.
19. M. Cova, C. Kruegel, and G. Vigna, "Detection and analysis of drive-by-download attacks and malicious javascript code," in *Proceedings of the 19th international conference on World wide web*, ser. WWW '10. New York, NY, USA: ACM, 2010, pp. 281–290. [Online]. Available: http://doi.acm.org/10.1145/1772690.1772720
20. M. Egele, P. Wurzinger, C. Kruegel, and E. Kirda, "Defending browsers against drive-by downloads: Mitigating heap-spraying code injection attacks," in *Detection of Intrusions and Malware, and Vulnerability Assessment*, ser.

Lecture Notes in Computer Science, U. Flegel and D. Bruschi, Eds. Springer Berlin Heidelberg, 2009, vol. 5587, pp. 88–106. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-02918-9_6

21. J. Oberheide, E. Cooke, and F. Jahanian, "CloudAV: N-version antivirus in the network cloud," in *USENIX Security Symposium*, P. C. van Oorschot, Ed. USENIX Association, 2008, pp. 91–106. [Online]. Available: http://www.usenix.org/events/sec08/tech/full_papers/oberheide/oberheide.pdf
22. I. Gashi, B. Sobesto, V. Stankovic, and M. Cukier, "Does malware detection improve with diverse antivirus products? an empirical study," in *Computer Safety, Reliability, and Security*, ser. Lecture Notes in Computer Science, F. Bitsch, J. Guiochet, and M. Kaniche, Eds. Springer Berlin Heidelberg, 2013, vol. 8153, pp. 94–105. [Online]. Available: http://dx.doi.org/10.1007/978-3-642-40793-2_9
23. http://www.nsslabs.com/.
24. http://www.amtso.org/.
25. T. Dumitras and D. Shou, "Toward a standard benchmark for computer security research: the worldwide intelligence network environment (wine)," in *Proceedings of the First Workshop on Building Analysis Datasets and Gathering Experience Returns for Security*, ser. BADGERS '11. New York, NY, USA: ACM, 2011, pp. 89–96. [Online]. Available: http://doi.acm.org/10.1145/1978672.1978683