

A Closer Look at Attack Clustering

Rainer Böhme

Institute for System Architecture
Technische Universität Dresden
rainer.boehme@tu-dresden.de

Gaurav Kataria

Heinz School of Policy and Management
Carnegie Mellon University
gauravk@andrew.cmu.edu

WORKING PAPER

Abstract

Worms cause correlated failure of many systems in a short span of time. Therefore, automated defensive approaches have been proposed to counter growth of worms. However, in addition to worms, many other kinds of cyber-attacks also exhibit significant correlation, albeit with slightly different properties. We argue that those specific correlation properties manifest because of the interaction between the attacker and the defender strategies. We survey the design space of defensive approaches and observe the extent of clustering (correlation) in attacks that these approaches are likely to induce. We highlight the implications of attack clustering on individual firms deploying these various approaches and also on global actors like government and cyber-insurance providers. We use 19 months of honeynet attack data to estimate clustering for some non-worm type attacks.

1 Introduction

Almost everyone agrees that we need better information security because there are so many threats and attacks happen too frequently. In this paper, we propose to look at another aspect of information security which has less to do with *how many* attacks there are but more to do with *how correlated* they are. Redundancy and distributed computing can to a large extent solve the problem of frequent failure of isolated systems. This is because if there are a million system failures distributed uniformly over the course of a year, then continued operation of most systems can be ensured on a daily basis by appropriate capacity planning, on the contrary, if average failure rate is still million a year but all of them fail at once on a single day, then in spite of some redundancy continued operation of all critical systems cannot be ensured. Computer worms are often credited with such spiked failures [53].

There is considerable research in the area of developing automated defenses to contain worm outbreaks [15, 24–26, 29, 30, 42, 43, 52]. However, worms are not the only threat to information systems, we believe that many worm defense approaches are too narrow in their context and fail to appreciate the underlying attack life cycle, which is determined by the interaction of the demand and supply of vulnerable systems over time. The supply of vulnerable systems changes as new vulnerabilities are discovered and old vulnerabilities are removed (patched) or temporarily fixed. Attackers' capability to exploit vulnerabilities (which is akin to demand) changes with time as knowledge about those vulnerabilities diffuses within the attacker community. Therefore, the length as well as second order characteristics of the attack life cycle determines the correlation in failure/survival of systems worldwide. Consider an example, a worm can affect many systems worldwide until its signature is developed and distributed across the world, and if the software vulnerability that the worm had exploited is not patched soon (as is often the case), then a new variant of that worm can too affect many systems until its signature is developed and distributed. This trend, which is also true for non-worm type attacks, often continues and leads to long and

clustered attack cycles. In this paper, we discuss the factors that alter the attack life cycle in predictable ways, and the implications of the resulting failure correlation structures for individual firms seeking security and society at large. We provide an extensive survey of defensive approaches that have been proposed so far from automated signature generation to software diversification, and estimate their likely impact on the attack life cycle.

The goal of this paper is to look at attack life cycles of some recently released vulnerabilities to determine correlation or clustering of attacks as function of time, review past literature on vulnerability/attack life cycle, and provide an objective comparison of different defensive approaches with respect to correlation pattern that they may induce in the attack life cycle.

The rest of this paper is organized as follows. In section 2, we present the contemporary trends in cyber attacks and defenses, and develop a framework to model the attack life cycle. In section 3, we introduce the notion of clustered attacks and explain the importance of this notion in quantifying the benefit of various defensive approaches in a unified way. In section 4, we provide a general empirical way to measure correlation of attacks for fast growing (worm-like) as well as slowly propagating attacks. And assess the implications of attack correlation for reactive defensive approaches. We conclude in section 5, by summarizing our findings.

2 Cyber Attacks and Defenses

Figure 1 shows the usual vulnerability profile of a system. While some vulnerabilities are due to faulty software applications, others manifest due to lack of user training and knowledge. Users often do not realize the security implications of their otherwise innocuous actions. As a consequence, majority of attacks exploit user vulnerabilities: Symantec estimates that 92% of most malware related reports involve email, while 14% are P2P (peer-to-peer file sharing) related [14]. Software vulnerabilities form a much smaller chunk of only about 13% of all reports.

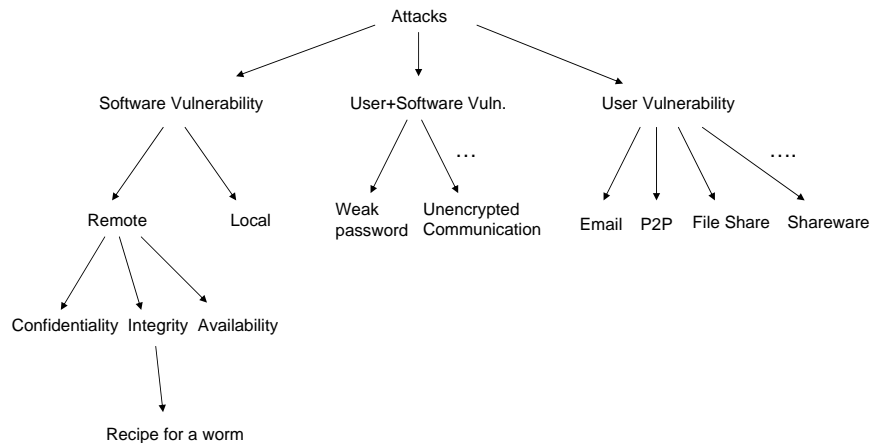


Figure 1: Classification of information security threats

While user vulnerabilities are addressed via awareness, training and desktop anti-virus software; software vulnerabilities require patches or temporary fixes. Worms—self propagating malware—capitalize on remotely exploitable software vulnerabilities. And while considerable attention has gone towards limiting worm infection by detecting and filtering malware not enough thought has been given to vulnerabilities in general and kinds of attack life cycle that they may induce due to the simple interaction between demand and supply of vulnerable systems.

Past literature on security incidents has found that attacks exploiting known vulnerabilities reoccur for extended periods of time. Arbaugh et al. [1] found that attacks continue to rise after the release of patches fixing software vulnerabilities and this trend can continue for months, while

the overall vulnerability attack life cycle lasts for years. Arora et al. [2] confirmed this observation using attack rate data from 2002–2003 for over three hundred different vulnerabilities. Beattie et al. [7] point out that firms need not apply security patches immediately after their release due to uncertainty about their reliability and concerns about system stability. In fact, they find that many patches are either pulled back or re-released, and on average it takes about 10–30 days for patches to get resolved after their initial release date. Therefore, firms often find it optimal to wait before applying patches, and large firms even conduct their own in-house testing of patches to address their stability concerns. While unpatched systems lie vulnerable for months, temporary defenses like firewalls are erected to block incoming malicious traffic.

The usual timeline of events with respect to patching and other defensive approaches is as follows:

1. Vulnerability/attack/worm detection — in most cases vulnerabilities are known well before attacks or worm(s) exploiting them appear.
2. Patch/signature generation — although patches or signatures may be generated in an automated fashion, testing for false positives takes a long time. Usually, the official patch developed by the vendor is better than temporary patches or fixes developed by others, because vendors have a more rigorous testing schedule. The process of testing a patch can take weeks.
3. Patch/signature dissemination — centralized dissemination takes few hours to days. P2P-based content distribution can speed up dissemination and algorithms have been proposed to secure the broadcast communication on Internet [46].
4. Patch/signature verification — usually large enterprises do not install a patch before they conduct their own thorough on-site testing. This process can take a week or more.
5. Installation — ideally installation should be instantaneous but due to factors like night and weekends, it may be delayed by a few hours to a day, and sometimes even more [38].

In order to shorten this long window of exposure, considerable attention in computer security literature has gone toward developing automated defenses [15,24,25,29,30,42,43,48,52], however, in this literature not enough thought has been given towards understanding the underlying concerns surrounding the business implications of information systems. For instance, temporary fixes like automatic patches and signatures that suffer from high false positive rates are less likely to be adopted in practice. Gartner [21] found that most companies do not find IDSes useful in practice, because of the costs associated with large false positives.

Some experts may argue that in today’s world all computers are set to automatically receive software patches and anti-virus updates, therefore, windows of exposure have already shrunk. We would like to point out that despite these advances, windows of exposure are still very long (sometimes more than a year) and are being actively exploited by attackers (see, Figure 2). Knowing that not all administrators/users patch their systems promptly, attackers continue to exploit a vulnerability as long as they can. Temporary fixes like firewall rules only “hide” the vulnerable systems from known exploits; variants of old exploits are developed by attackers to bypass these temporary defenses. In the following sub-section, we discuss how this game of cat and mouse leads to some predictable trends in the attack cycle.

2.1 Understanding the Attack Life Cycle

The number of vulnerable computers in the world, with respect to a particular vulnerability, is a decreasing function of time. When a vulnerability is first discovered, all systems (running that application) are vulnerable, but not necessarily attackable. With time, individuals and organizations patch their systems and thus number of vulnerable systems (with respect to a particular vulnerability) go down. On the other hand, attackers’ capability to attack/reach vulnerable systems increases with time.

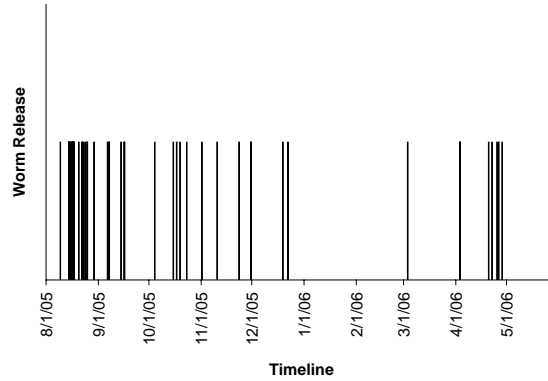


Figure 2: A timeline showing long attack cycle for worms exploiting vulnerabilities in MS05-039 (Windows Plug-n-Play). Vertical bars indicate release of worm on that date. Data Source: Symantec Corporation [12].

As time progresses defenders apply temporary fixes like firewall rules and IPS (intrusion prevention systems) to block or neutralize attacks that have been discovered by the defense community (Figure 3, left graph). For some time, these defenses successfully create a stricter upper bound on number of vulnerable systems reachable by attackers. However, attackers respond by developing variants of old exploits that evade defenses by either changing the signature or attack propagation vector. E.g. if a firewall filter starts blocking a particular network exploit then attackers may either modify the exploit or use a secondary channel like email or laptop to first get onto the internal network and then use the exploit to propagate on internal network.

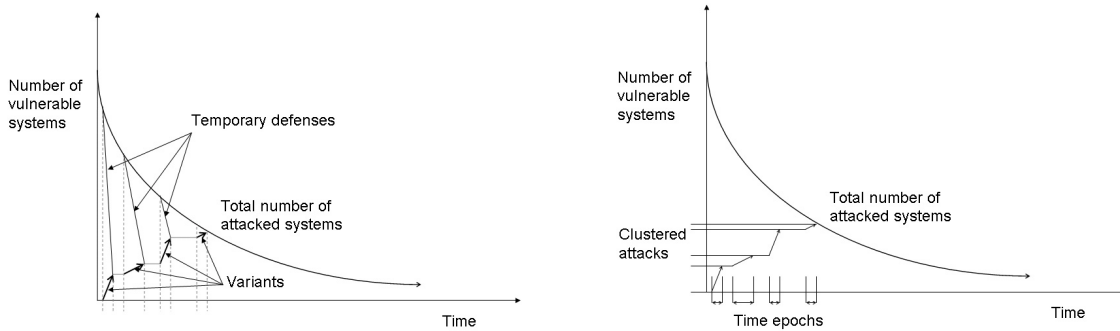


Figure 3: The dynamic interaction between attack and defense. *Left:* While the number of vulnerable systems decreases over time, the cumulative number of systems successfully attacked increases over time in sudden jumps with release of newer attack tactics. *Right:* The rate of successful attacks is highly variable over time, which leads to spikes (clustering) in attacks.

A particular variant can only affect systems till the time it is blocked. However, as new variants are developed over time, they are able to affect vulnerable systems which were previously protected by some temporary measures and were considered safe. On a time horizon we see many such instances when there is an outbreak followed by a period of lull, to be followed by another outbreak (Figure 3, right graph pictorially depicts that situation).

This trend of attack cycle is true for worms as well as other exploits that are not self-propagating, e.g. viruses, phishing, spam etc. We believe that researchers understand this trend, however, many of the security approaches that they have proposed have not fully factored this in.

2.2 Survey of Proposed Information Security Defenses

In this section we compare various information security approaches in terms of their practicality from the defenders point of view and the reaction they are likely to elicit from attackers.

In order to evaluate the effectiveness and viability of any proposed defensive approach, its role has to be seen the organizational context, which includes information assets it is trying to protect, other defensive approaches in place, the cooperation required from both within and outside the organization and the cost incurred in terms of deployment as well as management. We survey the many proposed approaches and attempt to evaluate them based on their suitability on the following three dimensions.

1. **Universality:** Range of attack types that can be tackled with the approach, i.e., how flexible is the (implicit) adversary model—worms, hacks, polymorphic exploits, multi-modal exploits etc.
2. **Accuracy:** Error rates, i.e. false positives and negatives.
3. **Cooperation-required:** To what extent does an approach require cooperation and trust from other parties involved, e.g. placing multiple sensors worldwide for collaborating and sharing information may warrant high trust requirements.

We group the approaches into representative classes and note for each class the likely impact they may have on attack correlation both within and across organizations. For example, all approaches that rely overly on perimeter security create a hard exterior and soft interior, causing high correlation of system failures within the organization, while maintaining a low correlation globally.

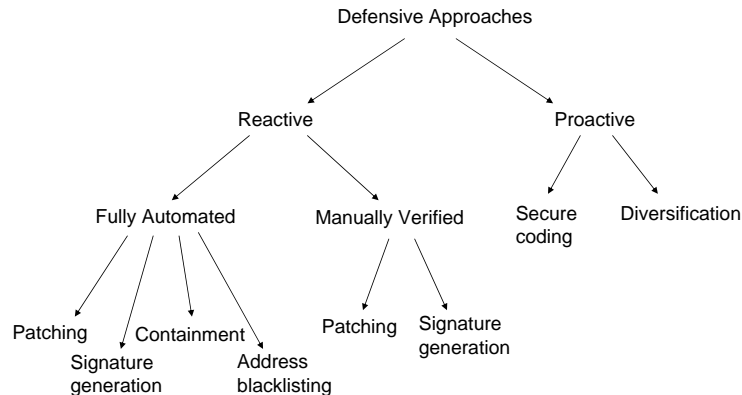


Figure 4: Classification of contemporary defense techniques

A broad classification of approaches is shown in Figure 4 and comparison of various classes is given in Table 1. Broadly speaking, we can say that proactive approaches are universal, reliable, centralized and integrate well with all other approaches. And among the reactive approaches, manual approaches though slower are more reliable because of supervised testing and verification, as compared to fully automated defenses. In general, the primary concern with fully automated defenses is their high false positive and false negative rates. We find that most papers in security literature that propose automated defenses do not report meaningful false positive rates (with some exceptions); they just say that it is “low”. Incremental research in this area has usually compared its performance to previous solutions, so the literature consists of a number of pair-wise comparisons. The lack of comparable benchmarks in this area is a detriment to the practicability

Table 1: Summary of survey on contemporary defensive techniques

Class	Universality	Accuracy	Cooperation	Attack Clustering
Conventional patching				
	Handles all kinds of exploits as it permanently removes the vulnerability.	The most accurate of all defensive approaches – supervised testing at vendor as well as on-site testing and verification.	Centralized development. All patches are developed and distributed by the vendor.	Has a long window of exposure [2, 14]. Cannot protect against exploits launched prior to the release of patch.
Automated patching [42, 46]				
	Same as above.	Accuracy is doubtful.	Due to the source code requirement only vendors can implement this approach.	Considerably reduces the window of exposure. Therefore, can reduce correlation in failure.
Manually verified signature generation [13, 33, 40]				
	Provides perimeter level security. Cannot protect against multimodal exploits that can penetrate through other channels.	Manual verification ensures that there are fewer false positives, but cannot sufficiently guard against new variants or polymorphic exploits.	Distributed deployment to gather exploit code and tactics. However, signatures are developed, verified and distributed by a single organization or a consortium.	Has a short but alternating window of exposure. Induces high on-site but low global correlation of attack success.
Fully automated signature generation [15, 24, 25, 30, 43, 52]				
	Same as above.	Has higher false positive rates. Context aware signatures will likely have higher accuracy than signatures generated by just pattern matching on malicious traffic.	Signatures can be generated either independently on-site, or can be collaboratively developed and distributed by participating sites.	Same as above. But it can reduce on-site correlation of failure by providing fail-safe mode of operation.
Containment through rate limiting [45, 49]				
	Is effective only against worm-like or fast propagating attacks.	Accuracy can be ensured by explicitly white-listing legitimate applications.	Very high level of cooperation required for it to be implemented globally.	Will likely reduce global correlation but will exhibit high on-site correlation if a particular site is affected.
Address blacklisting [26]				
	Is effective against only highly concentrated attack sources or for slowly propagating attacks.	The update frequency or refresh rate of these blacklists can severely affect their accuracy.	Very high level of cooperation required for it to be implemented globally.	Will likely reduce global correlation but will exhibit high on-site correlation if a particular site is affected.
Software diversification [5, 6, 19, 23, 32]				
	Handles all kinds of exploits as it reduces the impact of every vulnerability.	Does not remove vulnerabilities, it only reduces the probability of an attack being successful in exploiting a vulnerability.	Support for memory-layout and instruction set randomization to be provided by the vendor. Whereas, choice of software diversification largely lies with individual organizations.	Truly random deployment can very effectively reduce correlation of failure; however, if each site chooses to be internally homogeneous then it will induce high on-site correlation if a particular site is affected [9].
Secure coding (static and runtime analysis) [4, 10, 16, 17, 22, 28, 47]				
	Handles all kinds of exploits as it discovers the vulnerability at the time of inception/testing.	Imposes significant burden on developers to verify all the warning flags raised.	Completely centralized; either performed by the vendor voluntarily or imposed by a mandatory standard/certification.	Ideally, no exploit should exist.

of these approaches. We believe that these approaches are unlikely to be implemented unless there are thorough studies comparing their false positive rates to widely accepted benchmarks.

The most significant advantage of automated defenses is that signatures can be generated on-site and deployed in a short period of time [42]. Therefore, automatic signature/patch generation techniques that can generate signatures based on attacks received at a single site can protect other systems at that same site without cooperation from other sites. This aspect makes these techniques attractive for handling targeted attacks that may not be observed at all sites.

A limitation that is shared by both manual as well as automated signature generation techniques is that they require some level of decentralized data collection and sharing among multiple organizations. Gal-Or and Ghose [20] have shown that limited incentives exist for firms to share information regarding security related incidents. Moreover, concerns remain about the threat of adversaries penetrating the trusted data collection network [15, 46].

3 Clustered Attacks

In addition to universality, accuracy and cooperation requirement, we believe that every defensive approach has to be evaluated on the attack life cycle it is likely to induce (Table 1, column 5). Attack life cycles are generated by the strategic interaction between the attackers' and the defenders' techniques. It would be presumptuous to assume that attack life cycle can be completely eliminated. In this section, we look at the second order characteristics of the attack life cycle and its implications.

3.1 Implications of Attack Clustering

Clustered failure of information systems has significant implications for individual firms that plan their operating capacity assuming a fixed failure rate as well as for global stake holders like governments and insurance companies that certainly expect some isolated large-scale incidents but do not expect enormously-large-scale or catastrophic incidents. We present four specific cases where attack clustering matters.

3.1.1 Storage/backup Systems

Most backup systems store data based some *thresholding scheme* [37, 41, 50]. These schemes have three parameters: n , m and p (where $n \geq m \geq p$). The information asset of a firm is divided among n nodes. Assuming that some dependencies exist among them, at least p nodes need to be compromised to breach any useful information (where p can also be equal to 1 in case of no dependency). Due to presence of some redundancy in the system the entire information can be recreated with help of any m nodes, or information can be restored if number of node failures is no greater than $n - m$. Therefore, within a time period if less than $n - m$ nodes are breached, then information can be restored at the end of the period. Whereas, if there is high variance in failure due to attack correlation then in spite of average failure rate being same, information cannot be restored at the end of every period.

3.1.2 Automated Defense with Signature Generation

Although the goal of automated defenses is to reduce correlation of failure, to some extent they rely on the attack correlation itself to be effective. For example, signature generation techniques expect to observe multiple attacks at same or different sensors to develop reliable signatures [24, 25, 29, 30, 43]. In a way, the time to generate signature for an exploit is in itself dependent on rate of propagation of the exploit. Even those techniques that can reliably generate a signature after observing only one exploit have to wait until an attack is observed by at least one of the sensors. Therefore, such techniques though likely to effective against scanning based exploits are less likely to be effective against exploits that are successful in evading sensors. Therefore, the

only recourse for firms is to either implement very fast reacting filters or focus on applying patches as soon as possible to avoid being affected in first place.

3.1.3 Cyber-terrorism / Critical Infrastructure Protection

Many critical systems of great national and economic significance are present on Internet. An attack that can simultaneously takedown online banking or trading sites can have significant impact on economy. If a vulnerability is discovered in SCADA systems that control nation's critical infrastructure, then a foreign government or a terrorist organization can use that to simultaneously attack all vulnerable components. Although, in the literature clustered attacks have always been seen in the context of rapidly propagating worms, clustered attacks can also be launched by a resourceful adversary. Some of the automated techniques like signature generation and containment can only limit growth of a worm-type infection that relies on scanning to seek targets. The effectiveness of these techniques are very limited when highly targeted attacks are launched either through remotely controlled zombies (botnets) or through hit-list based worms. Moreover, a determined adversary can launch many variants of an exploit simultaneously in order to beat signature generation. Therefore, the possibility of a determined and resourceful adversary (foreign government) using a multi-modal and multi-variant approach to launch targeted attacks against critical infrastructure exists. The only defensive approach that holds promise in this respect is large scale software diversification such that vulnerabilities are (hopefully) not discovered simultaneously in all critical systems and it becomes very costly for an adversary to accumulate stocks of vulnerabilities secretly.

3.1.4 Cyber-insurance

The cost to an insurer that offers cyber-insurance can be written as:

$$C = E(L) + A + i \cdot c \quad (1)$$

Where, $E(L)$ is the expected loss amount, L being a random variable. A is the sum of all administrative costs. c is the safety capital required to settle all claims if the realization of L turns out to the ϵ -worst case (ϵ is the probability of ruin for the insurer). And i is the interest rate to be paid for the safety capital c , where it should reflect the risk associated with the business in general and the choice ϵ in particular [18]. In case of cyber-risks the random variable L is influenced by both attack correlation within an organization and across organizations. Risk-averse firms buy insurance to avoid carrying risk of correlated failure of their systems. Therefore, those classes of cyber risks that retain high level of intra-firm correlation are likely to be insured against by individual firms. On the other hand, those risk classes that exhibit high global attack clustering are likely to demand higher safety capital [8]. Considering these dynamics, we hypothesize that insurers are likely to promote research in techniques like automatic-signature generation as they hold promise for reducing global clustering while retaining local clustering. Though it may not seem obvious at first, but from the insurers' point of view the techniques like software diversification are preferable to stack randomization even if the average probability of failure is same under both, because former is expected to retain islands of homogeneity, whereas the latter will distribute vulnerable systems uniformly across the installed base.

4 Empirical and Experimental Evidence

In the previous sections we have argued that attack correlation is an important metric to look at in gaining macroscopic insights of attack activity on the Internet. This assertion, however, was based on theoretical considerations combined with intuition from anecdotal evidence and case studies. It is obvious that a thorough validation and development of reliable prediction model for historical (and eventually future) states cannot be covered in a single paper. Nevertheless, we use the remaining part of this paper to contribute a modest proposal of how attack clustering can be

measured on real data, which provides—though still suboptimal—a first strong support for the existence of attack clustering. We further demonstrate how models of the attack arrival process calibrated on real data can be employed in simulation studies to gain insights for the development of defensive approaches.

Before presenting our approach, we briefly acknowledge relevant literature that regards attack activity from a global perspective. Network telescopes have been used to infer trends on global attack patterns, especially DDoS attacks [27, 51]. Cooke et al. [11] have argued for distributed placement of sensors worldwide to compare attack trends. Numerous security consulting companies already operate sensors or collect security logs from their clients worldwide to characterize evolving attack trends [13].

4.1 Measuring Global Attack Correlation from HoneyNet Data

We use honeypot data to measure attack activity. Honeypots are dedicated hosts placed as decoys on the Internet. They simulate the interaction of vulnerable systems, by means of specialized honeypot software, and serve as monitoring devices for attack activities and exploit strategies [44]. One can distinguish between low and high interaction honeypots via degree of reactivity of the simulated system [36]. High interaction honeypots emulate an entire operating system and are reported to deceive even human attackers, whereas low interaction honeypots merely react to the first communication attempts and therefore serve primarily as trap for automated attacks. As the latter collect more standardized and better quantifiable data, we use data from the *Leurre.com* [35] honeynet project that runs dozens of low interaction honeypot sensors deployed at partner organizations worldwide.¹

The data is principally structured as event series, where records of type $(t, \mathcal{L}, \mathcal{S}, h)$ denote that sensor location \mathcal{L} has recognized $h > 0$ hits of port sequence \mathcal{S} at time t (measured in GMT calendar days). Port sequences consist of one or more ports in the TCP/IP protocol being targeted in a row from a unique source. Hits h are the absolute number of incoming connections from various sources in a day. For our analyses we use a boolean variable *attacks* instead of hits, where attack equal to 1 denotes a nonzero number of hits per unique combination of $(t, \mathcal{L}, \mathcal{S})$.

Port sequences have been reported as a simple and quite reliable indicator for identifying attack types [36], though there might remain some ambiguity about activity on popular ports, like port 80 used by HTTP. To reduce the level of such “noise” in our analysis we systematically researched vulnerabilities in the range of our data with uncommon port numbers. Though we believe that most packets sent to these ports are either attacks or precursor to attacks we cannot rule out possibility of mis-configured machines sending packets on these ports. Table 2 summarizes the vulnerabilities in our study. Care was taken to ensure that our sample had only few vulnerabilities per port and those could be attributed a single vendor product. By doing so, we were able to precisely isolate the attack life cycle of individual vulnerabilities, and correlate their characteristics. Moreover, the attack cycle on these ports were not influenced by release of any particular worm. Therefore, it allowed us to make more general observations regarding 2nd order attack characteristics.

For comparability we also include port 80 (HTTP), which is exposed to a high level of background noise. The entire data set comprises 211 K hits and 61.8 K attacks.

Immediate data analysis was impeded by some inconsistency of data across time. The challenge in data selection lied particularly in identifying sensor downtimes and marking these periods as missing values to diminish the risk of spurious correlation. Therefore, we applied the following data cleaning steps to generate our final data set for the subsequent correlation estimations:

1. Visually identified and selected the most dense time period of 19 months between September 2004 and March 2006.
2. The data was aggregated by total hits (including all port sequences) per sensor location and month. Sensor/month pairs with no hits were removed.

¹The raw data and the complete list of partners is kept confidential. Only affiliates to partner organization have access to aggregated data for research purpose.

Table 2: Overview of remote vulnerabilities under investigation

Service	Port	CVE	Disclosure	Patch
NNTP	119	2004-0574	8 Oct 2004	13 Oct 2004
WINS Service	42	2004-1080	26 Nov 2004	12 Dec 2004
Veritas backup server	6101	2004-1172	2 Nov 2004	16 Dec 2004
Veritas backup server	6101	2004-1389	11 Oct 2004	16 Dec 2004
Arkeia backup client	617	2005-0491,0496	20 Feb 2005	23 Feb 2005
CA license manager	10202	2005-0582,0583	8 Feb 2005	2 Mar 2005
Netvault backup	20031	2005-1009,1547	1 April 2005	5 May 2005
CA BrightStor backup	6050	2005-1018	14 Feb 2005	11 Apr 2005
CA BrightStor backup	6050	BID-12536	2 Dec 2004	11 Apr 2005
Veritas backup server	6106	2005-0771	18 Mar 2005	22 Jun 2005
Veritas backup client	10000	2005-0773,2079	16 Mar 2005	24 Jun 2005
CA BrightStor backup	6070	2005-1272	25 Apr 2005	2 Aug 2005
CA BrightStor backup	41523	2005-2535	14 Feb 2005	4 Aug 2005
Veritas backup client	10000	2005-2611	12 Aug 2005	12 Aug 2005
Veritas admin console	13722	2005-2715	12 Oct 2005	12 Oct 2005

Sorted by patch release date

- Sensor/month pairs where the number of days with at least one hit was below a threshold of 20 days were also removed. This step was aimed at eliminating months with partly up- and downtimes (e.g. some sensors may have gone online or offline multiple times within a month).
- Finally, data for sensors with less than 15 remaining months was removed. This step was essential to identify any time trends.

The so-reduced set of 67 % of the raw events comprises 17 sensors with relatively dense and homogeneous data. These sensor locations are distributed across 4 continents (Europe being slightly over-represented) and include partners in the IT industry, research institutions and telecommunication providers. Note that the downtime-removal is independent of the rare port sequences we selected. This contributes to carefully preventing spurious correlation.

To measure the correlation between locations across time we analyze the density function of the number of distinct sensor locations reporting at least one attack in a given time interval (e.g. one day). If the sensor locations were attacked independently then we would expect the number of attacked locations to follow a Binomial distribution (sum of Bernoulli trials with unconditional probability of attack π). A notable deviation from the Bernoulli distribution in turn is a clear sign of correlation. Figure 5 allows for a visual inspection of correlation: it depicts the total number of distinct attacked locations per day separately for each vulnerability in the study. It is very well visible that after an exploit appears it is merely a matter of a few days until a very large fraction of sensor locations recognized attacks *at the same day*.

Still on a visual basis, we can observe the relationship between the vulnerability disclosure date, the patch date, and the occurrence of the first attack activity. This is emphasized in figure 6, where the cumulative fraction of sensor locations having recognized at least one attack of a particular type is plotted relative to the disclosure date (left chart) and patch date (right chart). Apparently exploits for the backup systems (Arkeia, Netvault, and CA BrightStor) appeared *before* the public was informed about the existence of a problem whereas in case of WINS service attacks take off *after* a public vulnerability announcement. For CA license manager it appears as if the release of a patch has triggered the exploits. The situation is less clear for Veritas admin console and NNTP because disclosure and patch day fall close together. Finally, Veritas backup client and server have seen attacks long before the vulnerability release. Please note that this might as well be an artifact of other services using the same port (port 10000 is also used by Webmin, Zabbix,

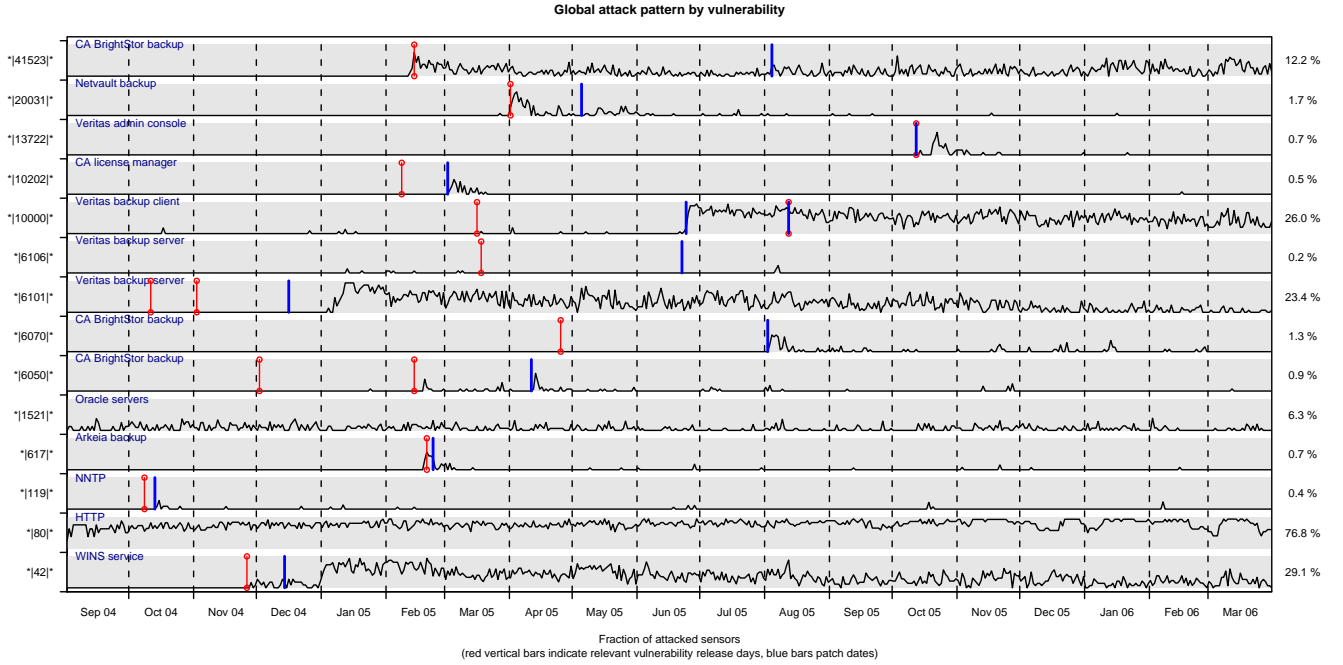


Figure 5: Fraction of distributed sensors that recognize at least one attack (daily data); leftmost list states port number; rightmost percentage figures indicate relative attack intensity by vulnerability. Vertical red lines (with two circles) indicate vulnerability disclosure dates and blue lines indicate patch release dates.

OpwinTrojan and by ndmp protocol in general).

The deviation from a Binomial distribution can also be quantified by applying a Beta-binomial model that allows for a dependency structure between the individual Bernoulli trials modeling the probability of attack at each sensor location. This model has already been used in system reliability literature to model correlated failure of backup systems [3] and to model failure across multiple versions of software [31]. The Beta-Binomial distribution is computed by randomizing the parameter π (probability of failure) of the Binomial distribution by a Beta distribution.

The probability that $X \sim \text{BB}(n, \pi, \rho)$ sensor locations face attack is given by

$$P(X = x|n, \pi, \rho) = \frac{B(n - x + \beta, x + \alpha)}{(n + 1) B(n - x + 1, x + 1) B(\alpha, \beta)} \quad (2)$$

$$\text{where } \alpha = \pi \cdot \left(\frac{1}{\rho} - 1\right), \quad \beta = (1 - \pi) \cdot \left(\frac{1}{\rho} - 1\right) \quad \text{and} \quad B(a, b) = \frac{\Gamma(a) \cdot \Gamma(b)}{\Gamma(a + b)}.$$

Γ denotes the Gamma function, B is the Beta function, and the parameters include the total number of active sensor locations n at a time, the (unconditional) probability of failure π , and the correlation measure ρ in the range 0 (no correlation) to 1 (perfect dependence). Different points in time are assumed to yield independent realizations of X , which concurs with the notion of a stationary multivariate Bernoulli process ($t_{\max} = 577$ data points). This distribution model belongs to the class of Generalized Additive Models for Location, Scale and Shape (GAMLSS) and can be fitted to data using maximum penalized likelihood² [39]. Note that censored (i.e., missing) data is implicitly accounted for by letting n_t depend on t . Table 3 reports the resulting estimates (including 95% confidence intervals, likelihood ratio test (LRT) for nonzero correlation and AIC goodness-of-fit measure) for different port sequences \mathcal{S} .

²We use a logistic link function for π and a square-root link for ρ to keep the parameters in reasonable ranges.

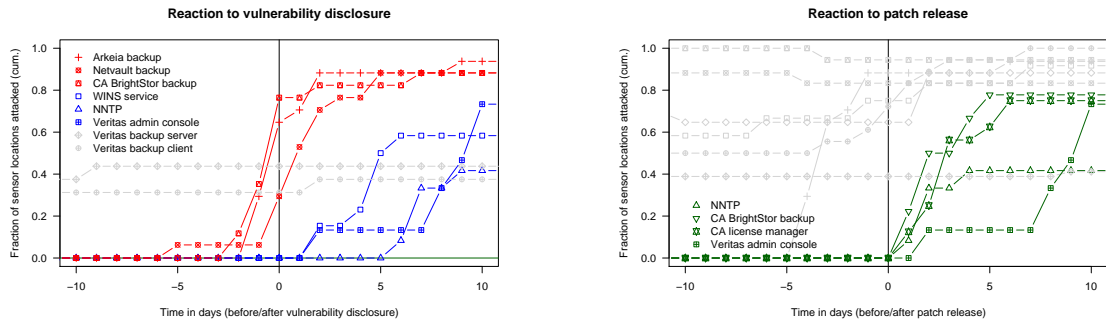


Figure 6: Fraction of sensor locations with attack attempts using of specified vulnerability. Cumulative measure over time relative to disclosure day (left) and patch day (right). The legend is common for both panels.

Table 3: Estimated correlation coefficients with Beta-binomial model

Vulnerability	Port	π	ρ	conf. int.	AIC	LRT
Oracle servers	1521	0.061	0.049	0.033–0.068	1490	$p_\alpha < 0.01$
Veritas backup server	6106	0.002	0.065	0.011–0.151	122	$p_\alpha < 0.01$
NNTP	119	0.003	0.086	0.024–0.172	200	$p_\alpha < 0.01$
CA BrightStor backup	6050	0.009	0.101	0.066–0.141	412	$p_\alpha < 0.01$
CA BrightStor backup	41523	0.124	0.128	0.104–0.152	2071	$p_\alpha < 0.01$
CA BrightStor backup	6070	0.013	0.193	0.132–0.256	481	$p_\alpha < 0.01$
WINS service	42	0.295	0.208	0.183–0.234	2829	$p_\alpha < 0.01$
Netvault backup	20031	0.018	0.228	0.170–0.287	570	$p_\alpha < 0.01$
Arkeia backup	617	0.007	0.237	0.133–0.343	274	$p_\alpha < 0.01$
CA license manager	10202	0.005	0.251	0.106–0.398	196	$p_\alpha < 0.01$
Veritas backup server	6101	0.232	0.277	0.245–0.308	2648	$p_\alpha < 0.01$
Veritas admin console	13722	0.007	0.293	0.167–0.414	262	$p_\alpha < 0.01$
Veritas backup client	10000	0.223	0.462	0.416–0.505	2498	$p_\alpha < 0.01$

$N = 577$, sorted by increasing ρ , confidence level 95 %

To back up our results and address the concern that the correlation is merely an artifact of the long period of “silence” before exploits are in the wild, we re-compute the estimation while excluding all data before the disclosure date. A comparison of this specification with the full data set is given in table 4. As expected, π is somewhat inflated and ρ somewhat underestimated if we use the post-disclosure sample. However, the effect is rather small and there are a number of exceptions where the relationship is inverse. Hence, we conclude that the correlation of attack is real. Which of the two measures is deemed as a better metric may depend on the application. Developers of automatic defense techniques may probably take the correlation after a specific vulnerability has been discovered as an exogenous condition in benchmarking scenarios, whereas actuaries underwriting cyber-insurance policies would prefer the unconditional measure of correlation or even an aggregation across multiple risk classes thereof.

It is interesting to observe that in the post disclosure case most backup systems observe strong correlation, which is even higher than standard Windows OS services like WINS and NNTP (IIS). Since the attack patterns were not influenced by release of any particular worm, we can make some general observations about the attackers’ response to these vulnerabilities. Specifically, the attackers may have expected that defenders who are guarding business information systems will apply patches (or block access) more quickly to their backup systems, than ordinary users would do to patch WINS service, therefore, they acted in a more concerted manner to attacks backup systems than other services. Though it is not possible for us to test this hypothesis immediately,

Table 4: Comparison between full and reduced sample

Vulnerability	Attack probability π			Attack correlation ρ			N_{post}
	full sample	post disclosure	change (in %)	full sample	post disclosure	change (in %)	
Oracle servers	0.061	NA	NA	0.049	NA	NA	NA
Veritas backup server	0.002	0.001	-48.7	0.065	0.241	272.2	379
NNTP	0.003	0.004	5.3	0.086	0.087	1.6	540
CA BrightStor backup	0.009	0.010	16.2	0.101	0.103	1.9	485
CA BrightStor backup	0.124	0.169	36.4	0.128	0.078	-38.8	411
CA BrightStor backup	0.013	0.022	66.2	0.193	0.190	-1.3	341
WINS service	0.295	0.346	17.6	0.208	0.145	-30.2	491
Netvault backup	0.018	0.027	54.1	0.228	0.230	0.7	365
Arkeia backup	0.007	0.009	30.1	0.237	0.231	-2.7	405
CA license manager	0.005	0.006	35.4	0.251	0.256	1.9	417
Veritas backup server	0.232	0.249	7.3	0.277	0.261	-5.6	537
Veritas admin console	0.007	0.025	241.3	0.293	0.275	-6.2	171
Veritas backup client	0.223	0.358	60.9	0.462	0.334	-27.6	381

$N_{\text{full}} = 577$, sorted by increasing ρ_{full}

an in-depth game theoretic modeling may provide better insight.

4.2 Simulating the Implications of Attack Clustering on Defense Techniques

In this section we demonstrate how calibrated models of attack activity might help in gaining insight regarding the effectiveness of various defensive techniques. Consider the evaluation of a defensive technique with respect to its ability to prevent the ϵ -worst possible outcome in terms of fraction of nodes failed simultaneously. Depending on the application, this scenario could arise when CIOs decide about the redundancy in a corporate IT environment, insurance supervisors gauge the likelihood of ruin, or policy makers assess the investment in critical infrastructure protection. Then the *reaction time* of the defensive technique to newly discovered threats is a critical parameter which is, for predictable risk classes, measurable between alternative systems. If we assume that shorter reaction times imply higher costs then the question of an optimal reaction time can be answered by estimating the associated level of attack correlation ρ as a function of reaction time from historical data. While it is evident that for independent attacks the number of hosts affected is linear in size of the exposure window, the dynamics of viral contagion of malware and reactivity to perimeter defenses are difficult to predict with purely analytical models. Moreover, attack clustering may also depend on the exposure in a non-linear way and thus add an additional adverse effect when the number of hosts attacked at the same time exceeds a certain application-specific bound (e.g., parameters m for integrity and p for confidentiality in storage systems).

To get a first insight about the relation between the reaction time and the degree of attack clustering for the cases in our sample we estimate parameter ρ for different time horizons. Analytically this is done by applying a pre-aggregation step while taking special care for the missing values in the data set. The results are given in table 5 for reaction times of 1, 2 and 3 days (to simulate automated techniques) as well as 7, 15 and 30 days (like in conventional patching). While the fact that correlation increases with the horizon is quite intuitive, it is rather remarkable that the slopes differ considerably between vulnerabilities. Note that these results should be regarded as broad assessment and it would be too far-fetched to interpret the individual figures from a simplistic model that has been calibrated on data of 17 sensors and 13 vulnerabilities only. The application of better methods with more elaborate models would require to make vulnerabilities as units of

Table 5: Simulation of attack correlation for different time horizons

Estimation for ρ	Time horizon					
	1 day	2 days	3 days	7 days	15 days	30 days
Oracle servers	0.05	0.07	0.07	0.08	0.12	0.11
Veritas backup server	0.06	0.06	0.10	0.09	0.17	0.27
NNTP	0.09	0.10	0.11	0.15	0.15	0.13
CA BrightStor backup	0.10	0.11	0.13	0.20	0.29	0.28
CA BrightStor backup	0.13	0.18	0.23	0.35	0.46	0.50
CA BrightStor backup	0.19	0.22	0.25	0.35	0.39	0.46
WINS service	0.21	0.26	0.29	0.33	0.33	0.34
Netvault backup	0.23	0.25	0.28	0.33	0.41	0.47
Arkeia backup	0.24	0.24	0.29	0.35	0.34	0.35
CA license manager	0.25	0.34	0.46	0.58	0.59	0.76
Veritas backup server	0.28	0.35	0.37	0.43	0.47	0.48
Veritas admin console	0.29	0.35	0.41	0.43	0.52	0.66
Veritas backup client	0.46	0.56	0.60	0.62	0.58	0.50

$N_1 = 577, N_2 = 289, N_3 = 193, N_7 = 83, N_{15} = 39, N_{30} = 20$

analysis instead of time, which demands a larger sample of vulnerabilities that are unambiguously identifiable in the data. The combination of our estimation techniques with heuristic approaches that classify attack types from multiple criteria [34] remains a promising target for future research.

5 Conclusion

In this paper we have postulated attack clustering is an important phenomenon to observe. We have done this by analyzing the *reasons* for attack clustering from a differentiated view on the mechanisms behind the attack and defense in information security. In particular, we have proposed a stylized model of the attack life cycle, which has been discussed in the context of variety of proposed defensive techniques. Further, we have addressed the *consequences* of attack clustering on technical, economic as well as governmental areas of interest. Finally, we have proposed and demonstrated a metric to *measure* the level of correlation for a selected type of risk class, where we found evidence for the existence of attack clustering in honeynet data.

References

- [1] W. A. Arbaugh, W. L. Fithen, and J. McHugh. Windows of vulnerability: A case study analysis. *IEEE Computer*, 33(12):52–59, 2000.
- [2] A. Arora, R. Krishnan, A. Nandkumar, R. Telang, and Y. Yang. Impact of vulnerability disclosure and patch availability- an empirical analysis. In *Workshop on the Economics of Information Security (WEIS)*, Minneapolis, MN, 2004.
- [3] M. Bakkaloglu, J. Wylie, C. Wang, and G. Ganger. On correlated failures in survivable storage systems, 2002. Technical Report CMU-CS-02-129, Carnegie Mellon University, School of Computer Science.
- [4] A. Baratloo, N. Singh, and T. Tsai. Transparent run-time defense against stack smashing attacks. In *Proceedings of the USENIX Annual Technical Conference*, 2000.
- [5] E. Barrantes, D. Ackley, S. Forrest, T. Palmer, D. Stefanovic, and D. Zovi. Randomized instruction set emulation to disrupt binary code injection attacks. In *Proceedings of the 10th ACM Conference on Computer and Communications Security (CCS '03)*, 2003.

- [6] E. G. Barrantes, D. H. Ackley, T. S. Palmer, D. Stefanovic, and D. D. Zovi. Randomized instruction set emulation to disrupt binary code injection attacks. In *CCS '03: Proceedings of the 10th ACM conference on Computer and communications security*, pages 281–289, 2003.
- [7] S. Beattie et al. Timing the application of security patches for optimal uptime. In *Proceedings of LISA 2002: 16th Systems Administration Conference*, pages 233–242, Berkeley, CA, 2002. USENIX Association.
- [8] R. Böhme and G. Kataria. Models and measures for correlation in cyber-insurance. In *Workshop on the Economics of Information Security (WEIS)*, University of Cambridge, UK, 2006. <http://weis2006.econinfosec.org/docs/16.pdf>.
- [9] P.-Y. Chen, G. Kataria, and R. Krishnan. Software diversity for information security. In *Workshop on the Economics of Information Security (WEIS)*, Harvard University, Cambridge, MA, 2005. <http://infosecon.net/workshop/pdf/47.pdf>.
- [10] M. Christodorescu and S. Jha. Static analysis of executables to detect malicious patterns. In *Proceedings of the 12th USENIX Security Symposium*, August 2003.
- [11] E. Cooke, M. Bailey, Z. M. Mao, D. Watson, F. Jahanian, and D. McPherson. Toward understanding distributed blackhole placement. In *WORM '04: Proceedings of the 2004 ACM workshop on Rapid malware*, 2004.
- [12] S. Corporation.
- [13] S. Corporation. Enterprise early warning solutions. <http://enterprisesecurity.symantec.com/SecurityServices/content.cfm?ArticleID=1522>.
- [14] S. Corporation. *Symantec Internet Security Threat Report, Volume IX*. 20330 Stevens Creek Road, Cupertino CA, 2006.
- [15] M. Costa, J. Crowcroft, M. Castro, A. I. T. Rowstron, L. Zhou, L. Zhang, and P. T. Barham. Vigilante: end-to-end containment of internet worms. In *SOSP*, pages 133–147, 2005.
- [16] C. Cowan, M. Barringer, S. Beattie, and G. Kroah-Hartman. FormatGuard: Automatic protection from printf format string vulnerabilities. In *Proceedings of the 10th USENIX Security Symposium*, August 2001.
- [17] C. Cowan, C. Pu, D. Maier, J. Walpole, P. Bakke, S. Beattie, A. Grier, P. Wagle, Q. Zhang, and H. Hinton. StackGuard: Automatic adaptive detection and prevention of buffer-overflow attacks. In *Proc. 7th USENIX Security Conference*, pages 63–78, jan 1998.
- [18] P. Embrechts, C. Klüppelberg, and T. Mikosch. *Modelling Extremal Events for Insurance and Finance*. Springer Verlag, Berlin Heidelberg, second edition, 1999.
- [19] S. Forrest, A. Somayaji, and D. Ackley. Building diverse computer systems. In *HOTOS '97: Proceedings of the 6th Workshop on Hot Topics in Operating Systems (HotOS-VI)*, 1997.
- [20] E. Gal-Or and A. Ghose. The economic incentives for sharing security information. *Information Systems Research*, 16(2):186–208, 2005.
- [21] Gartner Inc. Hype cycle for information security, May 2003. http://www.gartner.com/5_about/press_releases/pr11june2003c.jsp.
- [22] T. Jim, G. Morrisett, D. Grossman, M. Hicks, J. Cheney, and Y. Wang. Cyclone: A safe dialect of c, 2002.
- [23] G. S. Kc, A. D. Keromytis, and V. Prevelakis. Countering code-injection attacks with instruction-set randomization. In *Proceedings of the 10th ACM conference on Computer and communications security (CCS '03)*, 2003.

- [24] H. Kim and B. Karp. Autograph: Toward automated, distributed worm signature detection. In *Proceedings of the 13th USENIX Security Symposium*, August 2004.
- [25] C. Kreibich and J. Crowcroft. Honeycomb – Creating intrusion detection signatures using honeypots. In *Proceedings of the Second Workshop on Hot Topics in Networks (HotNets-II)*, November 2003.
- [26] D. Moore, C. Shannon, G. M. Voelker, and S. Savage. Internet quarantine: Requirements for containing self-propagating code. In *INFOCOM*, 2003.
- [27] D. Moore, G. M. Voelker, and S. Savage. Inferring internet Denial-of-Service activity. In *Proceedings of the 10th USENIX Security Symposium*, August 2001.
- [28] G. C. Necula, S. McPeak, and W. Weimer. CCured: type-safe retrofitting of legacy code. In *Symposium on Principles of Programming Languages*, pages 128–139, 2002.
- [29] J. Newsome, B. Karp, and D. Song. Polygraph: Automatic signature generation for polymorphic worms. In *Proceedings of the IEEE Security and Privacy Symposium*, May 2005.
- [30] J. Newsome and D. Song. Dynamic taint analysis for automatic detection, analysis, and signature generation of exploits on commodity software. In *Proceedings of the 12th Annual Network and Distributed System Security Symposium (NDSS '05)*, February 2005.
- [31] V. F. Nicola and A. Goyal. Modeling of correlated failures and community error recovery in multiversion software. *IEEE Transactions on Software Engineering*, 16(3):350–359, 1990.
- [32] A. J. O'Donnell and H. Sethu. On achieving software diversity for improved network security using distributed coloring algorithms. In *ACM Conference on Computer and Communications Security*, pages 121–131, 2004.
- [33] V. Paxson. Bro: A system for detecting network intruders in real-time. In *Proceedings of the 7th Usenix Security Symposium*, January 1998.
- [34] F. Pouget and M. Dacier. Honeypot-based forensics. In *AusCERT2004, AusCERT Asia Pacific Information technology Security Conference 2004, 23rd - 27th May 2004, Brisbane, Australia*, May 2004.
- [35] F. Pouget, M. Dacier, and V. H. Pham. Leurre.com: On the advantages of deploying a large scale distributed honeynet platform. In *Proc. of E-Crime and Computer Conference (ECCE)*, Monaco, March 29–30 2005. http://www.honeynet.org/papers/individual/ECCE_pouget_dacier_pham.pdf.
- [36] F. Pouget and T. Holz. A pointillist approach for comparing honeypots. In K. Julisch and C. Krügel, editors, *Detection of Intrusions and Malware, and Vulnerability Assessment (DIMVA 2005)*, LNCS 3448, pages 51–68, Berlin Heidelberg, 2005. Springer Verlag.
- [37] M. O. Rabin. Efficient dispersal of information for security, load balancing and fault tolerance. *Journal of the ACM*, 32(2):335–348, 1989.
- [38] E. Rescorla. Security holes . . . who cares? In *Proceedings of the 12th USENIX Security Symposium*, August 2003.
- [39] B. Rigby and M. Stasinopoulos. Generalized additive models for location, scale and shape. *Applied Statistics*, 54:507–554, 2005.
- [40] M. Roesch. Snort – Lightweight intrusion detection for networks. In *Proceedings of the 13th Systems Administration Conference (LISA '99)*, November 1999.
- [41] A. Shamir. How to share a secret. *Communications of the ACM*, 22(11):612–613, 1979.

- [42] S. Sidiroglou and A. D. Keromytis. Countering network worms through automatic patch generation. In *IEEE Security & Privacy*, pages 41–49, 2005.
- [43] S. Singh, C. Estan, G. Varghese, and S. Savage. Automated worm fingerprinting. In *Proceedings of the 6th ACM/USENIX Symposium on Operating System Design and Implementation (OSDI)*, December 2004.
- [44] L. Spitzer. *Know your enemy: Revealing the Security Tools, Tactics, and Motives of the Blackhat Community*. Addison-Wesley, 2001.
- [45] J. Twycross and M. Williamson. Implementing and testing a virus throttle. In *12th USENIX Security Symposium*, 2003.
- [46] M. Vojnovic and A. J. Ganesh. On the effectiveness of automatic patching. In *Proceedings of 3rd Workshop on Rapid Malcode (WORM)*, Nov 2005.
- [47] D. Wagner, J. S. Foster, E. A. Brewer, and A. Aiken. A first step towards automated detection of buffer overrun vulnerabilities. In *NDSS*, 2000.
- [48] H. J. Wang, C. Guo, D. R. Simon, and A. Zugenmaier. Shield: vulnerability-driven network filters for preventing known vulnerability exploits. In *SIGCOMM*, pages 193–204, 2004.
- [49] N. Weaver, S. Staniford, and V. Paxson. Very fast containment of scanning worms. In *Proceedings of the 13th Usenix Security Symposium*, pages 29–44, 2004.
- [50] J. J. Wylie et al. Survivable information storage systems. *IEEE Computer*, 33(8):61–68, 2000.
- [51] V. Yegneswaran, P. Barford, and J. Ullrich. Internet intrusions: Global characteristics and prevalence. In *In Proceedings of ACM SIGMETRICS, June 2003.*, 2003.
- [52] V. Yegneswaran, J. T. Giffin, P. Barford, and S. Jha. An architecture for generating semantics-aware signatures. In *Proceedings of USENIX Security Symposium*, 2005.
- [53] C. C. Zou, W. Gong, and D. Towsley. Code red worm propagation modeling and analysis. In *Proceedings of the 9th ACM conference on Computer and communications security*, pages 138 – 147. ACM, November 2002.