The Work-Averse Cyber Attacker Model: Theory and Evidence From Two Million Attack Signatures

Luca Allodi[a], Fabio Massacci[c,*], Julian Williams[b]

[a] *Department of Mathematics and Computer Science,*
Eindhoven University of Technology, Eindhoven, The Netherlands.
[b] *Durham University Business School, Mill Hill Lane, Durham, UK.*
[c] *Department of Information Engineering and Computer Science, University of Trento, Trento, Italy.*

Abstract

A common conceit is that the typical cyber attacker is assumed to be all powerful and able to exploit all possible vulnerabilities with almost equal likelihood. In this paper we present, and empirically validate, a novel and more realistic attacker model. The intuition of our model is that a mass attacker will optimally choose whether to act and weaponize a new vulnerability, or keep using existing toolkits if there are enough vulnerable users. The model predicts that mass attackers may i) exploit only one vulnerability per software version, ii) include only vulnerabilities with low attack complexity, and iii) be slow at introducing new vulnerabilities into their arsenal. We empirically test these predictions by conducting a natural experiment for data collected on attacks against more than one million real systems by Symantec's WINE platform. Our analysis shows that mass attackers fixed costs are indeed significant and that substantial efficiency gains can be made by individuals and organizations by accounting for this effect.

Keywords: Cyber Security, Dynamic Programming, Malware Production, Risk Management
JEL Classification: C61, C9, D9, L5

Vulnerabilities in an information systems allow cyber attackers to exploit these affected systems for financial and/or political gain. Whilst a great deal of prior research has focused on the security investment decision making process for security vendors and targets, the choices of attackers are less well understood. This paper is the first attempt to directly parameterize cyber attacker production functions from first principles and the empirically fit this function to empirical data.

Our main results challenge the notion of all powerful attackers able to exploit a broad range of security vulnerabilities and provides important guidance to policy makers and potential targets on how to utilize finite resources in the presence of cyber threats.[1]

[*] Corresponding Author. The authors thank Matthew Elder at Symantec Corp. for his many useful comments. A preliminary treatments of this dataset and conceptual idea have been presented at the 2015 European Conference on Information Systems (ECIS 2015). The authors retain copyright for future publication purposes. This research has been partly supported from the European Union's Seventh Framework Programme for research, technological development and demonstration under grant agreement no 285223 (SECONOMICS), from the Italian PRIN Project TENACE, and from the EIT Digital Master School 2016 - Security and Privacy programme. Our results can be fully reproduced by recovering the reference data set *WINE-2012-008*, archived in the WINE infrastructure.

Email addresses: `l.allodi@tue.nl` (Luca Allodi), `fabio.massacci@unitn.it` (Fabio Massacci), `julian.williams@durham.ac.uk` (Julian Williams)

[1] An early security reference on this conceit can be found in (Dolev and Yao 1983a, page 199) where a protocol should be secured "against arbitrary behavior of the saboteur". The Dolev-Yao model is a quasi "micro-economic" security model: given a population with several (interacting) agents, a powerful attacker will exploit any weaknesses, and security is violated if she can compromise some honest agent. So, *every* agent must be secured by mitigating *all* vulnerabilities. A more gentle phrasing suggests that the likelihood that a given vulnerability is exploited should be at maximum entropy and hence the only dominating factor will be the criticality of that vulnerability.

May 29, 2017

A natural starting point when attempting to evaluate the decision making of attackers is to look at the actual 'traces' their attacks leave on real systems (also referred as '*attacks in the wild*'): each attempt to attack a system using a vulnerability and an exploit mechanism generates a specific attack signature, which may be recorded by software security vendors. Dumitras and Shou (2011) and Bilge and Dumitras (2012) provide a summary of signature identification and recording, whilst Allodi and Massacci (2014) shows how they can be linked to exploit technology menus. By observing the frequency with which these attack signatures are triggered, it is possible to estimate (within some level of approximation) the rate of arrival of new attacks. Evidence from past empirical studies suggests a different behavior depending on fraud type; for example, Murdoch et al. (2010) shows that attackers focussing on chip and pin credit cards, which require physical access, are very proactive and rapidly update their menu of exploits; for web users, Allodi and Massacci (2014) and Nayak et al. (2014) indicate that the actual risk of attacks in the wild is limited to hundred vulnerabilities out of the fifty thousand reported in vulnerability databases. Mitra and Ransbotham (2015) confirm these findings by showing that even (un)timely disclosures do not correlate with attack volumes.

These empirical findings are at odds with the classical theoretical models of attacker behaviour by which attackers can and will exploit any vulnerability Dolev and Yao (1983a), and remain largely un-explained from a theoretical perspective. This paper fills this gap by identifying a new attacker model that realistically unifies the attacker's production function with empirical evidence of rates of arrival of new attacks worldwide. The current corpus of results provide strong prima-facie evidence that attackers do not quickly mass develop new vulnerability exploits that supplement or replace previous attack implementations. This collective behavior is at odds with the classical theoretical models of attacker behaviour Dolev and Yao (1983b): attackers should exploit *any* vulnerability. We must therefore conclude that attackers are rational, that the effort required to produce an exploit and hence deployable malware is costly, and that they will respond to incentives in a way that is consistent with classical models of behaviour (Laffont and Martimort 2009). Finally, this work directly impacts the development of risk models for cyber-attacks by identifying empirical and theoretical aspects of attack and malware production.[2]

Figure 1 shows the fractions of systems receiving attacks recorded by Symantec, a large security vendor, for two different cases: the red line plots the fraction of systems receiving two attacks at two different times that target the same software vulnerability (CVE). The abscissa values represent the time, in days, between attacks, hence we would expect that the red line would decrease (which it does) from near unity to zero. The black dashed line represents the opposite case: the same system and the same software are attacked but the attacker uses a new vulnerability, different from the original attack. The attack data suggests that it takes more than two years before the number of attacks using new vulnerabilities exceeds the number of attacks using the same vulnerabilities, and about 3-4 years before a

[2] Whilst challenging the maximum entropy notion may initially appear to the informed reader as attacking a straw man, the underpinning ideas of Dolev-Yao persist into to the internet era (through all phases), see for instance comments in Schneier (2008) that cover similar ground. Variants of the all-powerful attackers are proposed (e.g. honest-but-curious, game-based provable security models) but they only changed the power and speed of attacks not the will: if there is weakness that the attacker can find and exploit, they will. Papers analyzing web vulnerabilities Stock et al. (2013), Nikiforakis et al. (2014) report statistics on the persistence of these vulnerabilities on internet sites as evidence for this all powerful effect and broad coverage of security vulnerabilities. We would like to emphasize that we are not arguing, necessarily, that there is systematic over-investment in information security, but that presuming the probability mass function relating likelihood of a successful attack to exploitable vulnerabilities, by severity category, is at or close to maximum entropy is sub-optimal for security investment decision making.

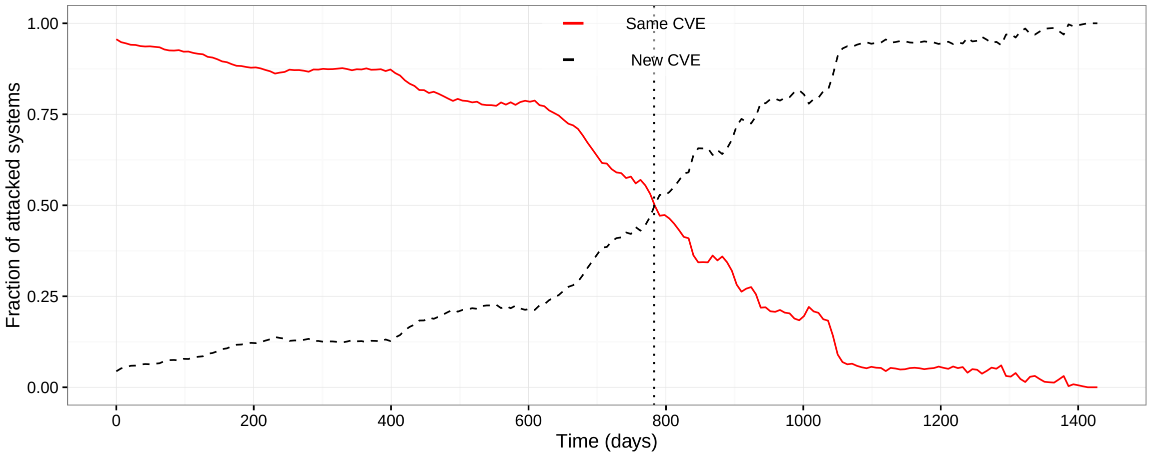

Figure 1: Distribution of time between of subsequent attacks with similar signatures.
Note: Fraction of systems receiving the same attack repeatedly in time (red, solid) compared to those receiving a second attack against a different vulnerability (black, dashed). The vertical line indicates the number of days after the first attacks where it becomes more likely to receive an attack against a new vulnerability rather than against an old one.

complete refresh occurs.

Our methodological contribution is twofold. First, we specify a novel theoretical model of the dynamic decision making process of the attacker that is based on Stokey's logic of inaction, see Stokey (2008). We model the timing of effort by the attacker as a dynamic programming problem, which we initially solve in its generality. For the purpose of empirical analysis we then restrict it to the case of an attacker focusing on the 'next' update.

Our main prediction is that the complexity of implementation of a vulnerability endogenously interacts with the update time and attackers will predominantly flock to the low hanging fruit of vulnerabilities with low complexity and high impact. Second, we capture such relations by deriving, directly from the theoretical model, a corresponding robust parametric regression model of equilibrium update times and hence reverse engineer part of the malware production function, a first in this literature. To ensure that the causal relations predicted in the theoretical model are captured in our empirical study, we control for several factors related to the characteristics of the user and their system for each recorded pair of attacks in the data (e.g. user geographical locations). This work is the first to explicitly consider an attacker with fixed-costs, and to validate the theoretical predictions with an empirical analysis derived directly from the analytical model.

The remainder of this paper is organized as follows §(1) provides a brief review of the current research on attacker technologies and production functions. §(2) provides a dynamic model of attacking effort in continuous time with discrete update intervals, this model is then used to compute a linear equilibrium between updates of attackers technologies and the intensity (measured by machines attacked). We utilize the specific insights from this model to determine the Markov conditions needed to estimate an empirical model in §(3) utilizing over two million attack signatures and the the vulnerabilities targeted by those signatures. We then provide some brief commentary on the results in §(4) and then outline implications and directions for future work in §(5).

1. Background

The economic decision making and the organization of information system security has been explored from a target perspective quite thoroughly in recent years. Anderson (2008) provides a good summary on the early extant literature.[3] Commonly, the presence of vulnerabilities has been considered as the starting point for the analysis, as it *could* allow an attacker to take (total or partial) control of a system and subvert its normal operation for personal gain.

Asghari et al. (2013), Van Eeten and Bauer (2008), Van Eeten et al. (2010) address the economic incentives (either perverse or well aligned) in addressing security threats. Market based responses, such as bug bounty programs, are discussed in Miller (2007) and Frei et al. (2010) whilst Karthik Kannan (2005) and Finifter et al. (2013) also address the issue of vulnerability markets and the value of vulnerabilities to developers, organizations that may potentially be targeted, and indeed attackers.

The consensus in the security literature appears to have settled on the view that the severity of the vulnerability, as measured by a series of metrics, should be used as a direct analogue of risk. For a broad summary of the metrics and risk assessments involved in this domain see Mellado et al. (2010) (technology based metrics); Sheyner et al. (2002), Wang et al. (2008), Manadhata and Wing (2011) (attack graphs and surfaces); and Naaliel et al. (2014), Wang et al. (2009), Bozorgi et al. (2010), Quinn et al. (2010) (severity measure metrics and indexation). From an economic perspective Ransbotham and Mitra (2009) studies the incentives that influence the diffusion of vulnerabilities and hence the opportunities of the attacker to attack a target's systems (software and infrastructure). The standard de facto metric for the assessment of vulnerability severity is the Common Vulnerability Scoring System, or CVSS[4] in short CVSS-SIG (2015). Hence, the view is that any open vulnerability will eventually be exploited by some form of malware and the target organization will then be subject to a cyber attack.

Recent studies in the academic literature have challenged the automatic transfer of the technical assessment of the 'exploitability' of a vulnerability into actual attacks against end users. Bozorgi et al. (2010) and Allodi and Massacci (2014) have empirically demonstrated on different samples a substantial lack of correlation between the observed attack signatures in the wild and the CVSS type severity metrics. The current trend in industry is to use these values as proxies demanding immediate action (see Beattie et al. (2002) for operating system security, PCI-DSS (2010) for credit card systems and Quinn et al. (2010) for US Federal rules).

In particular, prior work suggests that only a small subset of vulnerabilities are actually exploited in the wild (Allodi and Massacci 2014), and that none of the CVSS measures of severity of impact predict the viability of the vulnerability as a candidate for an implemented exploit (Bozorgi et al. 2010). Bozorgi et al. (2010) argue that besides the 'technical' measure of the system's vulnerabilities, other factors should be considered such as the value or cost of a vulnerability exploit and the ease and cost with which the exploit can be developed and then deployed.

Mitra and Ransbotham (2015) also indicate that early disclosure has no impact on attack volume (number of recorded attacks) and there is only some correlation between early dis-

[3] Policy, cost sharing and incentives have also been comprehensively explored from a target perspective in August and Tunca (2006, 2008, 2011) and Arora et al. (2004, 2008).

[4] The CVSS score provides a standardized framework to evaluate vulnerability severity over several metrics, and is widely reported in public vulnerability databases such as the National Vulnerability Database NIST (2015) maintained by the National Institute of Standards in Technology (NIST).

closure and the ‘time-to-arrival’ of exploits: hence providing additional evidence reinforcing the lack of ‘en mass’ migration of attacks to newly disclosed vulnerabilities.

On a similar line, Herley (2013) posits the idea that for a (rational) attacker not all attack types make sensible avenues for investment. This is supported by empirical evidence showing that attack tools actively used by attackers embed only ten to twelve exploits each on the maximum (Kotov and Massacci 2013, Grier et al. 2012), and that the vast majority of attacks recorded in the wild are driven by only a small fraction of known vulnerabilities (Nayak et al. 2014, Allodi 2015). It is clear that for the attacker some reward must be forthcoming, as the level of costly effort required to implement and deliver the attack observed in the wild is far from negligible. For example, Grier et al. (2012) uncovers the presence of an underground market where vulnerability exploits are rented to attackers (‘exploitation-as-a-service’) as a form of revenue for exploit writers. Liu et al. (2005) suggest that attacker economic incentives should be considered when thinking about defensive strategies: increasing attack costs or decreasing revenue may be effective in deterring the development and deployment of an attack. For example, Chen et al. (2011) suggests that a possible mean to achieve this is to ‘diversify’ system configurations within an infrastructure so that the fraction of attackable system by a single exploit diminishes, hence lowering the return for any given attack.

The effort needed to engineer an attack can be generally characterized along the three classic phases from Jonsson and Olovsson (1997): *reconnaissance* (where the attacker identifies potential victims), *deployment* (the engineering phase), *refinement* (when the attack is updated). The first phase is covered by the works of Wang et al. (2008), Howard et al. (2005), Nayak et al. (2014), where the attacker investigates the potential pool of targets affected by a specific vulnerability by evaluating the attack surface of a system, or the ‘popularity’ of a certain vulnerable software. The engineering aspects of an exploit can be understood by investigating the technical effort required to design one (see for example Schwartz et al. (2011) and Carlini and Wagner (2014) for an overview of recent exploitation techniques). However,the degree of re-invention needed to update an exploit and the anticipated time from phase two to phase three remain largely un-investigated (Yeo et al. 2014, Serra et al. 2015, Bilge and Dumitras 2012, provide some useful results in this direction).

Our model aggregates deployment and reconnaissance costs into a single measure, whereas we explicitly model the expected time to exploit innovation and subsequent updating times to include new vulnerabilities.

2. A Dynamic Attacker Model with Costly Effort

We consider a continuous time setting, such that $0 < t < \infty$, where an attacker will be choosing an optimal update sequence $0 < T_1 < T_2 < \ldots T_i \ldots T_\infty$ for weaponizing new vulnerabilities $v_1, .., v_n$. The attackers technology has a “combination” of exploit technology that undergoes periodic updates. Each combination targets a specific mix of vulnerabilities and we presume that the attacker can make costly investments to develop the capability of their technology.

The attacker starts their development and deployment activity at time $t = 0$ by initially identifying a set of vulnerabilities $V \subset \mathcal{V}$ from a large universe $\mathcal{V}$ affecting a large number of target systems N. A fraction θ_V of the N systems is affected by V and would be compromised by an exploit in absence of security countermeasures. Targets are assumed to deploy patches and/or update systems, whilst security products update their signatures (e.g. antiviruses, firewalls, intrusion prevention systems). Hence, the number of infected systems available for exploit will decay with time.

We can represent vulnerability patches and security signatures as arriving on users’ systems following two independent exponential decay processes governed respectively by the rates λ_p and λ_{sig}. The effect of λ_p and λ_{sig} on attacks has been previously discussed by Arora et al. (2004, 2008), Chen et al. (2011), whilst a discussion on their relative magnitude is provided in Nappa et al. (2015). Assuming that the arrival of patches and antivirus signatures are independent processes, and rolling them up into a single factor $\lambda = f(\lambda_p, \lambda_{sig})$, the number of systems impacted by vulnerabilities in V at time t is

$$N_V(t) = N\theta_V e^{-\lambda t}. \tag{1}$$

For the given set of vulnerabilities Vtargeted by their technologies combination the attacker will pay an upfront cost $C(V|\emptyset)$ and has an instantaneous stochastic profit function of

$$\Pi_V(t) = [r(t, N_V(t), V) - c(t, V)]e^{-\delta t}, \tag{2}$$

where $r(t, N_V, V)$ is a stochastic revenue component[5], whilst $c(t, V)$ is the variable costs of maintaining the attack[6], subject to a discount rate of δ. We do not make any assumption on the form that revenues take from successful attacks. They could be kudos in specific fora (Ooi et al. 2012) or revenues from trading the victim's assets in underground markets (Grier et al. 2012, Allodi et al. 2015).

At some point, the attacker might decide to perform a refresh of its attacking capabilities by introducing a new vulnerability and engineering its exploit by incurring an upfront cost of $C(v|V)$. This additional vulnerability will produce a possibly larger revenue $r(t, N_{V\cup\{v\}}(t), V \cup \{v\})$ at an increased marginal cost $c(t, V \cup \{v\})$. As the cost of engineering an exploit is large with respect to maintenance ($C(v|V) \gg c(t, V \cup \{v\})$) and neither successful infection (Allodi et al. 2013), nor revenues are guaranteed (Herley and Florencio 2010, Rao and Reiley 2012, Allodi et al. 2015), the attacker is essentially facing a problem of deciding action vs inaction in presence of fixed initial costs as described by Stokey (2008) and their best strategy is to deploy the new exploit only when the old vulnerabilities no longer guarantee a suitable expected profit.

This decision problem is then repeated over time for n newly discovered vulnerabilities, and n refresh times denoted by T_i.

We define by $C_0 = C(V|\emptyset)$ the initial development cost and by $C_{i+1} \equiv C(v_{i+1}|V \cup \{v_1 \dots v_i\})$ the cost of developing the new exploits, given the initial set V and the additional vulnerabilities $v_1 \dots v_i$. We denote by $N_i(t) \equiv N_{V\cup\{v_1,\dots,v_i\}}(t)$ the number of systems affected by adding the new vulnerability at time t. Similarly, we define $r_i(t)$ and $c_i(t)$ as the respective revenue and marginal cost of the vulnerability set $V \cup \{v_1, \dots, v_i\}$. Then the attacker faces the following stochastic programming problem for $n \to \infty$

$$\{T_1^*, \dots, T_n^*\} = \underset{\{T_1,\dots,T_n\}}{\arg\max} \sum_{i=0}^{n} -C_i e^{-\delta T_i} + \int_{T_i}^{T_{i+1}} \left(r_i(t, N_i(t)) - c_i(t)\right) e^{-\delta t} d\omega. \tag{3}$$

The action times $T_0 = 0$, $T_{i+1} > T_i$, and T_{n+1} is such that $r_n(T_{n+1}, N_n(T_{n+1})) = c_n(T_{n+1})$. Since the maintenance of malware, for example through 'packing' and obfuscation (i.e. techniques that change the aspect of malware in memory to avoid detection), is minimal and does

[5] This component accounts for the probability of establishing contact with vulnerable system (Franklin et al. 2007), the probability of a successful infection given a contact (Kotov and Massacci 2013, Allodi et al. 2013), and the monetization of the infected system (Kanich et al. 2008, Zhuge et al. 2009, Rao and Reiley 2012).

[6] For example, the attacker may need to obfuscate the attack payload to avoid detection (Grier et al. 2012), or renew the set of domain names that the malware contacts to prevent domain blacklisting (Stone-Gross et al. 2009).

not depend on the particular vulnerability, see (Brand et al. 2010, § 3) for a review of the various techniques, we have that $c_i(t) \to 0$ and therefore also $T_{n+1} \to \infty$. This problem can be tractably solved with the techniques discussed in Stokey (2008), Birge (2010) and further developed in Birge and Louveaux (2011) either analytically or numerically by simulation. Nevertheless, it is useful to impose some further mild assumptions that result in solutions with a clean set of predictions that motivate and place in context our empirical work in the standard Markovian set-up needed to identify specific effects.

By imposing the history-less (adapted process) assumption on the instantaneous payoff, with a risk neutral preference, the expected pay-off and expected utility for a given set of ordered action times $\{T_1, \ldots, T_n\}$ coincides. Risk preferences are therefore encapsulated purely in the discount factor, a common assumption in the dynamic programming literature (see the opening discussion on model choice in (Stokey 2008, Ch. 1) for a review).

Following (Birge and Louveaux 2011, Ch. 4) the simplest approach is to presume risk neutrality (under the discount factor δ) and solve in expectations as a non-stochastic Hamilton–Jacobi–Bellman type problem along the standard principles of optimal decision making.

Under the assumption of stationary revenues, we define r as the average revenue across all systems. The instantaneous expected time t payoff from deployed malware is approximated by the following function:

$$\mathbb{E}[r(t, N_{V\cup\{v\}}(t))] \doteq rN \left(\theta_V e^{-\lambda t} + (\theta_{V\cup\{v\}} - \theta_V)e^{-\lambda(t-T)}\right), \tag{4}$$

where $t \geq T$ is the amount of time since the attacker updated the menu of vulnerabilities (by engineering new exploits) at time T. The first addendum caters for the systems vulnerable to the set V of exploited vulnerabilities that have been already partly patched, whilst the second addendum accounts for the new, different systems that can be exploited by adding v to the pool. For the latter systems, the unpatched fraction restarts from one at time T.

Solving Eq. (3) in expectations we can replace the stochastic integral over $d\omega$ with a traditional Riemann integral over dt and evaluate the approximation in expectations. By solving the above integral and imposing the usual first order condition we obtain the following decomposition for the optimal profit for the attacker.

$$\{T_1^*, \ldots, T_n^*\} = \underset{\{T_1,\ldots,T_n\}}{\arg\max} \sum_{i=0}^{n} (\Pi(T_{i+1}, T_i) - C_i)e^{-\delta T_i} \tag{5}$$

$$\Pi(T_{i+i}, T_i) = \frac{rN}{\lambda+\delta}\left(\theta_i - \theta_{i-1} + \theta_{i-1}e^{-\lambda T_i}\right)\left(1 - e^{-(\lambda+\delta)(T_{i+1}-T_i)}\right) \tag{6}$$

where we abbreviate $\theta_{-1} \equiv 0$, $\theta_0 \equiv \theta_V$, and $\theta_i \equiv \theta_{V\cup\{v_1\ldots v_i\}}$.

Proposition 1. *The optimal times to weaponize and deploy new exploits for attackers aware of initial fixed costs of exploit development are obtained by solving the following n equations for $i = 1, \ldots, n$*

$$\frac{\partial\Pi(T_i, T_{i-1})}{\partial T_i}e^{-\delta T_{i-1}} - \delta(\Pi(T_{i+1}, T_i) - C_i)e^{-\delta T_i} + \frac{\partial\Pi(T_{i+1}, T_i)}{\partial T_i}e^{-\delta T_i} = 0 \tag{7}$$

subject to $T_0 = 0$, $T_{n+1} = \infty$, $\delta, \lambda > 0$.

Proof of Proposition 1 is given in Appendix 6.1. □

Unrestricted dynamic programming problems such as that described in Proposition 1 typically do not generally have analytic solutions for all parameter configurations. They can be log-solved either in numerical format, or by computer algebra as a system of n equations by setting $x_i = e^{-\lambda T_i}$, $y_i = e^{-\delta T_i}$ and then adding the n equations $\delta \log x_i = \lambda \log y_i$. However, we can relax the solution procedure to admit a myopic attacker who only considers a single individual decision $n = 1$. This approximates the case when the true dynamic programming problem results in $T_1^* > 0$ and $T_2^* \to \infty$. For an overview of the appropriate domains for the use of this type of simplification see DeGroot (2005).

Corollary 1. *A myopic attacker, who anticipates an adapted revenue process from deployed exploits subject to a decreased effectiveness due to patching and anti-virus updates with a negligible cost of maintenance for each exploit, will postpone indefinitely the choice of weaponizing a vulnerability v if the ratio between the cost of developing the exploit and the maximal marginal expected revenue is larger than the discounted increase in the fraction of exploited vulnerabilities, namely $\frac{C(v|V)}{rN} > \frac{\delta}{\lambda+\delta}(\theta_{V\cup\{v\}} - \theta_V)$. The attacker would be indifferent to the time at which deploy the exploit only when the above relation holds at equality.*

Proof of Corollary 1 is given in Appendix 6.2. □

Whilst our focus is on realistic production functions for the update of malware, it should be noted that the 'all–powerful' attacker is still admitted under our solution if the attacker cost function $C(v|V)$ for weaponizing a new vulnerability collapses to zero. In this case, Corollary 1 predicts that the attacker could essentially deploy the new exploit at an arbitrary time $[0, +\infty]$ even if the new exploit would not yield extract impact ($\theta_{V\cup\{v\}} = \theta_v$).

If the vulnerability v affects a software version for which there is already a vulnerability in V,the fraction of systems available for exploit will be unchanged ($\theta_v = \theta$). Hence, the cost has to be essentially close to zero ($C(v|V) \to 0$) for the refresh to be worth. In the empirical analysis section §(4) we report a particular case where we observe this phenomenon occurring in our dataset. Based on these considerations we can now state our first empirical prediction.

Hypothesis 1. Given Corollary 1, a work-averse attacker will overwhelmingly use only one reliable exploit per software version.

Engineering a reliable vulnerability exploit requires the attacker to gather and process technical vulnerability information (Erickson 2008).

This technical 'exploit complexity' is captured by the `Attack Complexity` metric provided in the CVSS. When two vulnerabilities cover essentially the same population ($\theta_{V\cup\{v\}} - \theta_V \approx \epsilon$) a lower cost would make it more appealing for an attacker to refresh their arsenal as this would make it easier to reach the condition ($\frac{C(v|V)}{rN} \approx \frac{\delta}{\lambda+\delta}(\theta_{V\cup\{v\}} - \theta_V) \approx \epsilon$) when the attacker would consider deploying an exploit to have a positive marginal benefit.

Hypothesis 2. Corollary 1 also implies that a work-averse attacker will preferably deploy low-complexity exploits for software with the same type of popularity.

Corollary 1 describes a result in the limit and in presence of a continuous profit function. Indeed according to Eq. (4) the attacker expects to make a marginal profit per unit of time equal to $rNf(t)$ where $\lim_{t\to\infty} f(t) \to 0$ and as a result $\frac{\partial\Pi(T_{i+1},T_i)}{\partial T_{i+1}}$ is a monotone decreasing function and $\frac{\partial\Pi(T_{i+1},T_i)}{\partial T_{i+1}} \to 0$ for $T_{i+1}b \to \infty$. In practice, the profit expectations of the attacker are discrete: as the marginal profit drops below r, it is below the expect marginal profit per unit of compromised computers. Hence, the attacker will consider the time $T_{i+1} = T^\star < \infty$ where such event happens as equivalent to the event where the marginal profit goes to zero ($T_{i+1} = \infty$) and hence assumes that the maximal revenue has been achieved and a new exploit can be deployed.

Proposition 2. *A myopic attacker, who anticipates an adapted revenue process from deployed exploits subject to a decreased effectiveness due to patching and anti-virus updates with a negligible cost of maintenance for each exploit, and expects a marginal profit at least equal to the marginal revenue for a single machine ($\frac{\partial \Pi}{\partial T} \geq \frac{r(0,N_V(0),V)}{N_V(0)}$) will renew their exploit at*

$$T^{\star} \;=\; \frac{1}{\delta}\log\left(\frac{C(v|V)}{r} - \frac{\delta}{\lambda+\delta}(\theta_{V\cup\{v\}} - \theta_V)N\right) \tag{8}$$

under the condition that $\frac{C(v|V)}{rN} \geq \frac{1}{N} + \frac{\delta}{\lambda+\delta}(\theta_{V\cup\{v\}} - \theta_V)$.

Proof of Proposition 2 is given in Appendix 6.3. □

Assuming the cost and integral of the reward function over $[0, T_i^*]$ are measured in the same numèraire and approximately within the same order of magnitude, the model implies that the discount factor plays a leading role in determining the optimal time for the new exploit deployment, the term $\frac{1}{\delta}$ in Eq. (8). Typically the extant microeconomics literature (see Frederick et al. 2002) sets $\exp(\delta) - 1$ to vary between one and twenty percent. Hence, a lower bound on T_1^* would be $\approx [100, 400]$ when time is measured in days. This implies the following prediction:

Hypothesis 3. Given Proposition 2, the time interval after which a new exploit would economically dominate an existing exploit is large, $T_1^* > 100$ days.

3. Experimental Data Set

Our empirical dataset merges three data sources, these are:

The **National Vulnerability Database (NVD)** is the vulnerability database maintained by the US. Known and publicly disclosed vulnerabilities are published in this dataset along with descriptive information such as publication date, affected software, and a technical assessment of the vulnerability as provided by the CVSS. Vulnerabilities reported in NVD are identified by a Common Vulnerabilities and Exposures identifier (CVE-ID) that is unique for every vulnerability.

The **Symantec threat report database (SYM)** reports the list of attack signatures detected by Symantec's products along with a description in plain English of the attack. Amongst other information, the description reports the CVE-ID exploited in the attack, if any.

The **Worldwide Intelligence Network Environment (WINE)**, maintained by Symantec, reports attacks detected in the wild by Symantec's products. In particular, WINE is a representative, anonymized sample of the operational data Symantec collects from users that have opted in to share telemetry data (Dumitras and Shou 2011). WINE comprises attack data from more than one million hosts, and for each of them, we are tracking up to three years of attacks. Attacks in WINE are identified by an ID that identifies the attack signature triggered by the detected event. To obtain the exploited vulnerability we match the attack signature ID in WINE with the CVE-ID reported in SYM.

The data extraction involved three phases: (1) reconstruction of WINE users' attack history; (2) building the controls for the data; (3) merging and aggregating data from (1) and (2). Because of user privacy concerns and ethical reasons, we did not extract from the WINE dataset any potentially identifying information about its hosts. For this reason, it is useful to distinguish two types of tables: tables *computed* from WINE, namely intermediate tables with detailed information that we use to build the final dataset; and *extracted* tables,

Table 1: Variables included in our dataset

Variable	Description
$\mathtt{CVE}_{1,2}$	The identifier of the previous and the current vulnerability v exploited on the user's machine.
$\mathcal{T}$	The delay expressed in fraction of year between the first and the second attack.
$\mathcal{N}$	The number of detected attacks for the pair *previous attack, actual attack.*
$\mathcal{U}$	The number of systems attacked by the pair.
Compl	The Complexity of the vulnerability as indicated by its CVSS assessment. Can be either High, Medium or Low as defined by CVSS(v2) Mell et al. (2007).
Imp	The Impact of the vulnerability measured over the loss in Confidentiality, Integrity and Availability of the affected information. It is computed on a scale from 0 to 10 where 10 represents maximum loss in all metrics, and 0 represents no loss. Mell et al. (2007).
Day	The date of the vulnerability publication on the National Vulnerability Database.
Sw	The name of the software affected by the vulnerability.
Ver	The last version of the affected software where the vulnerability is present.
Geo	The country where the user system is at the time of the second attack.
Hst	The profile of the user or "host". See Table 2 for reference.
Frq	The average number of attacks received by a user per day. See Table 2.
Pk	The maximum number of attacks received by a user per day. See Table 2.

containing only aggregate information on user attacks that we use in this research. The full list of variables included in our dataset and their description is provided in Table 1.[7]

3.1. Understanding the Attack Data Records

Each row in our final dataset of tables extracted from WINE represents an "attack-pair" received by Symantec users. We are interested in the new vulnerability v whose exploit has been attempted after an exploit for V vulnerabilities have been already engineered. Hence, for every vulnerability v identified by $\mathtt{CVE}_2$ our dataset reports the aggregated number of attacked users ($\mathcal{U}$) and the aggregated volume of attacks ($\mathcal{N}$) on the vulnerability v which have previously ($\mathcal{T}$ days before) received an attack on some other vulnerability identified by $\mathtt{CVE}_1$.

Additional information regarding both attacked CVEs is extracted from the NVD: for each CVE we collect the publication date (Day), the vulnerable software (Sw), the last vulnerable version (Ver), and an assessment of the Compl of the vulnerability exploitation and of its Imp, as provided by CVSS (v2). At the time of performing the experiment we use the second revision of the CVSS standard.

A CVE may have more than one attack signature. This is not a problem in our data as we are concerned with the exploitation event, and not with the specific footprint of the attack. However, attack signatures have varying degrees of generality, meaning that they can be triggered by attacks against different vulnerabilities but follow some common pattern. For this reason, some signatures reference more than one vulnerability.

[7] Researchers interested in replicating our experiments can find NVD publicly available at http://nvd.nist.gov; SYM is available online by visiting http://www.symantec.com/security_response/landing/threats.jsp. The version of SYM and NVD used for the analysis is also available from the authors at anonymized_for_the_submission; the full dataset computed from WINE was collected in July 2013 and is available for sharing at Symantec Research Labs (under NDA clauses for access to the WINE repository) under the reference *WINE-YYYY-NNN*. In the online Appendix 7 we provide a full 'replication guide' that interested researchers may follow to reproduce our results from similar sources by Symantec or other security vendors.

Table 2: Values of the Hst, Frq, and Pk control variables for WINE users.

Hst	Description
STABLE	Host does not update and does not change country between attacks.
ROAM	Host's system is the same but it changed location.
UPGRADE	Host's system was upgraded without moving.
EVOLVE	Host both upgraded the system and changed location.

Frq/Pk	Description
LOW	#attacks ≤ 1.
MEDIUM	#attacks ≤ 10.
HIGH	#attacks ≤ 100.
VERYHIGH	#attacks ≤ 1000.
EXTREME	#attacks > 1000.
Frq	average × day
Pk	maximum × day

Table 3: Summary Excerpt from our dataset.

$\texttt{CVE}_1$	$\texttt{CVE}_2$	$\mathcal{T}$	$\mathcal{U}$	$\mathcal{N}$	Geo	Hst
CVE-2003-0533	CVE-2008-4250	83	186	830	IT	UPGRADE
CVE-2003-0818	CVE-2003-0818	146	1	1	US	ROAM
CVE-2003-0818	CVE-2009-4324	616	1	1	CH	EVOLVE
CVE-2003-0818	CVE-2009-4324	70	52	55	US	EVOLVE

Note: Each $< \texttt{CVE}_1, \texttt{CVE}_2, \mathcal{T}, \texttt{Geo}, \texttt{Hst}, \texttt{Frq}, \texttt{Pk} >$ tuple is unique in the dataset. We here omit Frq, Pk, and all the CVSS details of $\texttt{CVE}_1$ and $\texttt{CVE}_2$ for brevity. The column $\mathcal{N}$ reports the number of attacks against $\texttt{CVE}_2$ received $\mathcal{T}$ days after an attack against $\texttt{CVE}_1$ suffered by $\mathcal{U}$ systems characterised by a host profile Hst and located in Geo.

In this case, we have no means to know which of the vulnerabilities was effectively exploited by the attack. Out of 1,573 attack signatures, 112 involve more than one vulnerability; to avoid introducing counting errors on the number of attacks per CVE, we drop these attack signatures.

Each pair also includes additional information on the type of host that received the attack. We use this information to control the user's profile in terms of countries he or she connects to the Internet from (i.e. we trace whether the user moves geographically), and whether the system he or she operates on changes. Users with profiles that change in time may look different to the attacker, and may therefore be subject to different attacks and attack volumes (see Chen et al. (2011), Kotov and Massacci (2013), Grier et al. (2012), Baltazar (2011) for a discussion).

In particular, we report the host's geographical location (Geo), and a number of proxy measures of the 'proneness' of the host in receiving an attack. Table 2 reports the measured values for each dimension and their definition.

The Hst variable measures whether the host changed geographical region since the first attack happened (as this may affect their likelihood of receiving an attack), and whether they update their system in the observation period. Frq and Pk measure respectively the average and maximum number of attacks received per day by the host. We use them as proxy variables measuring the 'exposure' of a host to attacks. Thresholds have been chosen based on the distribution of attacks received per day by users in WINE.

Table 3 reports an excerpt from the dataset. Each row represents a succession of attack pairs. The columns $\texttt{CVE}_1$ and $\texttt{CVE}_2$ report respectively the CVE-ID of the attacked vulnerability in v and in the novel attack against V. The column $\mathcal{T}$ reports the time delay, measured in days, between the two attacks. The column $\mathcal{N}$ reports the overall number of attacks detected for $\texttt{CVE}_2$ after an attack against $\texttt{CVE}_1$; $\mathcal{U}$ reports the number of single systems receiving the same pair of attacks.

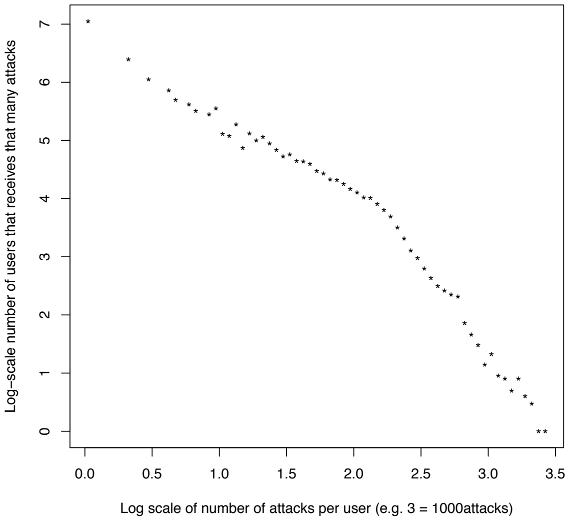

Figure 2: Distribution of attacks received per day by WINE users.
Note: Scatterplot distribution of attack frequency per user. Log-number of users is reported on the y-axis; the x-axis reports (log) frequency of attacks. There is an exponential relationship between the variables: few users receive thousands of attacks (> 1000 in the observed period), as opposed to the vast majority that only receive a few.

The column Geo reports the country in which the second attack was recorded. Finally, Hst reports the type of user affected by the attack as defined in Table 2. The same $\texttt{CVE}_i$ may appear multiple times in the dataset, as may the same pair $\texttt{CVE}_1, \texttt{CVE}_2$. For example the third and fourth rows in Table 3 reports attacks against $\texttt{CVE}_1 =$ CVE-2003-0818, $\texttt{CVE}_2 =$ CVE-2009-4324; one instance happened 616 days apart, while the other only 70. Only one WINE system in Switzerland (CH) suffered from this sequence of attacks, whilst 52 systems received this pair of attacks in the US. In both cases the systems considered are of type EVOLVE, indicating that the affected systems have been upgraded and moved from some other country to the country listed in Geo during our observation period.

Figure 2 reports the observed distribution on a logarithmic scale. It is apparent that most WINE hosts receive only a handful of attacks per day, while few hosts are subject to ‘extreme’ levels of attack density ($> 1,000/day$). These may be hosts subject to network floods sent by the attacker in a ‘scripted’ fashion, or a large number of individual users whose actual, private IP address is behind a large proxy (e.g. by their internet service provider).

3.2. Descriptive statistics

Descriptive statistics of all variables are reported in Table 4. Table 6 reports the count distribution of the number of WINE users for each of these factor’s levels. It is apparent that in the case of Hst users are uniformly distributed among the factor levels, with ROAM users being the least frequent category. Most of the mass of the Frq and Pk distributions is at the low end of the scale (i.e. most users receive few attacks per day both as an average and as a maximum). From Table 6 it appears that users characterized by EXTREME or VERYHIGH levels for Frq or Pk are outliers and may therefore need to be controlled for.

Finally, we evaluate the geographic distribution of attacks per user to identify geographical regions that may be more subject to attacks than others.

Figure 3 reports the distribution of the mean number of attacked systems per day in each geographic area of the five continents. The distributions are significantly different among and within continents with the exception of Africa, for which we observe little intra-continental

Table 4: Descriptive statistics of variables for $\mathcal{T}$, $\mathcal{N}$, $\mathcal{U}$, `Hst`, `Frq`, `Pk`, `Geo`.

Variable	Mean	St. dev.	Obs.	Variable	Mean	St. dev.	Obs.
Delay, Volume, Machines				`Geo`			
$\mathcal{T}$	0.99	0.831	2.57E+06	Australia & New Zeal.	0.039	0.194	1.01E+05
$\mathcal{N}$	13.552	102.28	2.57E+06	Caribbean	0.013	0.113	33,011
$\mathcal{U}$	11.965	93.11	2.57E+06	Central America	0.007	0.084	18,078
				Central Asia	0	0.005	63
`Hst`				Eastern Africa	0.001	0.026	1,733
EVOLVE	0.552	0.497	1.42E+06	Eastern Asia	0.043	0.202	1.09E+05
ROAM	0.109	0.312	2.80E+05	Eastern Europe	0.011	0.104	28,277
STABLE	0.076	0.265	1.95E+05	Melanesia	0	0.008	159
UPGRADE	0.263	0.441	6.77E+05	Micronesia	0.001	0.035	3,123
				Middle Africa	0	0.007	112
`Frq`				Not Avail.	0.112	0.316	2.89E+05
EXTREME	0.004	0.063	10,099	Northern Africa	0.002	0.04	4,088
HIGH	0.179	0.384	4.61E+05	Northern America	0.468	0.499	1.20E+06
LOW	0.436	0.496	1.12E+06	Northern Europe	0.023	0.151	60,009
MEDIUM	0.379	0.485	9.75E+05	Polynesia	0	0.003	28
VERYHIGH	0.001	0.037	3,572	South America	0.008	0.09	20,794
				South-Eastern Asia	0.023	0.15	59326
`Pk`				Southern Africa	0	0.02	1031
EXTREME	0	0.011	292	Southern Asia	0.013	0.114	33,901
HIGH	0.296	0.456	7.60E+05	Southern Europe	0.063	0.242	1.61E+05
LOW	0.091	0.288	2.35E+05	Western Africa	0.001	0.034	2,960
MEDIUM	0.609	0.488	1.57E+06	Western Asia	0.018	0.134	46,830
VERYHIGH	0.004	0.063	10,182	Western Europe	0.154	0.361	3.95E+05

Table 5: Descriptive statistics for $\texttt{CVE}_1$ and $\texttt{CVE}_2$ variables.

	$\texttt{CVE}_1$				$\texttt{CVE}_2$		
Variable	Mean	St. dev.	Obs.	Variable	Mean	St. dev.	Obs.
$\texttt{Compl}_{CVE1,H}$	0.009	0.094	22,769	$\texttt{Compl}_{CVE2,H}$	0.009	0.096	23,803
$\texttt{Compl}_{CVE1,L}$	0.42	0.494	1.08E+06	$\texttt{Compl}_{CVE2,L}$	0.334	0.472	8.58E+05
$\texttt{Compl}_{CVE1,M}$	0.571	0.495	1.47E+06	$\texttt{Compl}_{CVE2,M}$	0.657	0.475	1.69E+06
$\texttt{Imp}_{CVE1}$	9.549	1.417	2.57E+06	$\texttt{Imp}_{CVE2}$	9.681	1.37	2.57E+06
Internet Explorer	0.096	0.295	2.47E+05	Internet Explorer	0.04	0.196	1.03E+05
PLUGIN	0.791	0.407	2.03E+06	PLUGIN	0.9	0.3	2.31E+06
PROD	0.083	0.276	2.13E+05	PROD	0.037	0.189	95,404
SERVER	0.03	0.171	77,507	SERVER	0.023	0.149	58,634
Pub. Year	2008.8	2.231	2.57E+06	Pub. Year	2009.4	2.131	2.57E+06

variance. The highest mean arrival of attacks per day is registered in Northern America, Asia and throughout Europe with the exception of Southern Europe. The mapping of country and region is defined as in the World Bank Development Indicators.[8]

4. Empirical Analysis

The data in our sample is quite obviously unique and hence prior to conducting any correlative analysis we illustrate some scenarios that provide prima facie statistical evidence on the validity of the hypotheses identified from our theoretical model.

In accordance with Hypothesis 1 the attacker should prefer to (a) attack the same vulnerability multiple times rather than for only a short period of time, and (b) create a new exploit only when they want to attack a new software version.

To evaluate these scenarios we identify three types of attack pairs that are summarized in Table 7: in the first type of attack pair (A_1) the first attacks and the second attack affect the

[8] See `http://data.worldbank.org/country` for a full categorization and breakdown.

Table 6: Number of WINE users in the `Hst`, `Frq`, and `Pk` control groups.

`Hst`	#Users	`Frq`	#Users	`Pk`	#Users
`STABLE`	1,446,020	`LOW`	3,210,465	`LOW`	2,559,819
`ROAM`	96589	`MEDIUM`	311,929	`MEDIUM`	983,221
`UPGRADE`	306,856	`HIGH`	19,683	`HIGH`	3,783
`EVOLVE`	1,697,570	`VERYHIGH`	3,919	`VERYHIGH`	170
		`EXTREME`	1,039	`EXTREME`	42

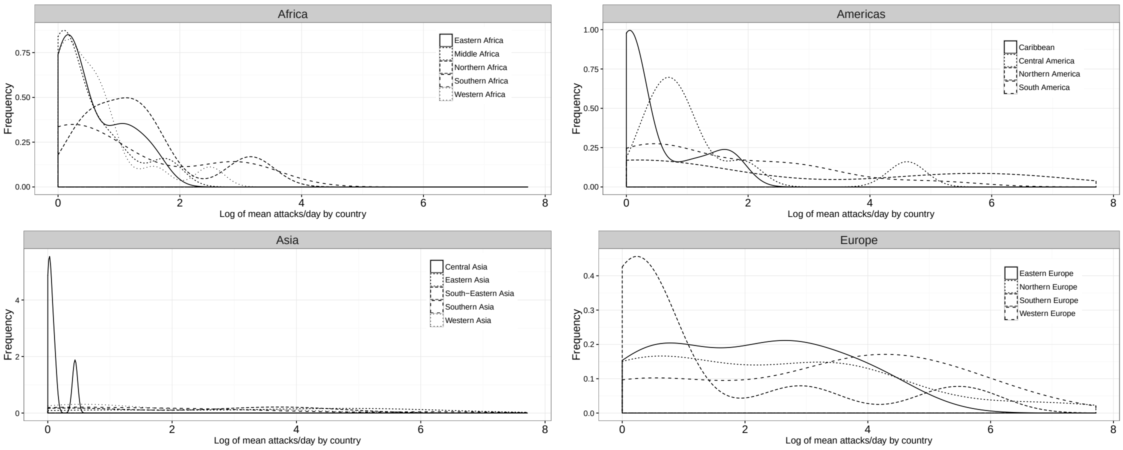

Figure 3: Distribution of attacks per day received in different geographic regions.
Note: Kernel density of mean cumulative attacks per day by geographical region. Regions in Americas, Asia, and Europe show the highest rates of attacks. Attack densities vary can vary substantially per region. Oceania is not reported because it accounts for a negligible fraction of attacks overall.

same vulnerability and, consequently, the same software version; in the second pair (A_2) the first attack and the second attack affect the same software, but different CVEs and different software versions; finally the first and second attacks affect the same software and the same version but exploit different vulnerabilities (A_3). According to our hypothesis we expect that A_1 should be more popular than A_2 (in particular when the delay between the attacks is small) whilst A_3 should be the least popular of the three.

To evaluate these attacks it is important to consider that users have diverging models of software security (Wash 2010), different software have different update patterns and update frequencies (Nappa et al. 2015), and different attack vectors (Provos et al. 2008).

For example, an attack against a browser may only require the user to visit a webpage, while an attack against a word processing application may need the user to actively open a file on the system (see also the definition of the `Attack Vector` metric in the CVSS standard CVSS-SIG (2015)). As these clearly require a different attack process, we further classify `Sw` in four categories: SERVER, PLUGIN, PROD(-ductivity) and Internet Explorer. The categories are defined by the software names in the database. For example SERVER environments are typically better maintained than 'consumer' environments and are often protected by perimetric defenses such as firewalls or IDSs. This may in turn affect an attacker's attitude toward developing new exploits. This may require the attacker to engineer different attacks for the same software version in order to evade the additional mitigating controls in place. Hence we expect the difference between A_2 and A_3 to be narrower for the SERVER category.

Figure 4 reports a fitted curve of targeted machines as a function of time by software

Table 7: Sample Attack Scenarios and Compatibility with Work-Aversion Hypothesis

Type	Condition	Description	Hypothesis
A_1	$\texttt{CVE}_1 = \texttt{CVE}_2$	The first attacks and the second attack affect precisely the same vulnerability and, consequently, the same software version	Often for Hyp 3 as $T^\star \to \infty$
A_2	$\texttt{CVE}_1 \neq \texttt{CVE}_2 \wedge$ $\texttt{Sw}_{\texttt{CVE}_1} = \texttt{Sw}_{\texttt{CVE}_2} \wedge$ $\texttt{Ver}_{\texttt{CVE}_1} \neq \texttt{Ver}_{\texttt{CVE}_2}$	The first attack and the second attack affect the same software but different CVEs and different software versions.	Less frequent for Hyp 1 and Hyp 2 as $0 < T^\star < \infty$
A_3	$\texttt{CVE}_1 \neq \texttt{CVE}_2 \wedge$ $\texttt{Sw}_{\texttt{CVE}_1} = \texttt{Sw}_{\texttt{CVE}_2} \wedge$ $\texttt{Ver}_{\texttt{CVE}_1} = \texttt{Ver}_{\texttt{CVE}_2}$	First and second attacks affect the same software and the same version but exploit different vulnerabilities	Almost never for Hyp 1 as $\theta_{V \cup \{v\}} = \theta_V$

Note: We expect the vast majority of attacks generated by the work-averse attacker to be of type A_1. A_2 should be less frequent than A_1, as it requires to engineer a new exploit. A_3 contradicts the work aversion hypothesis and should be the least common type of attack.

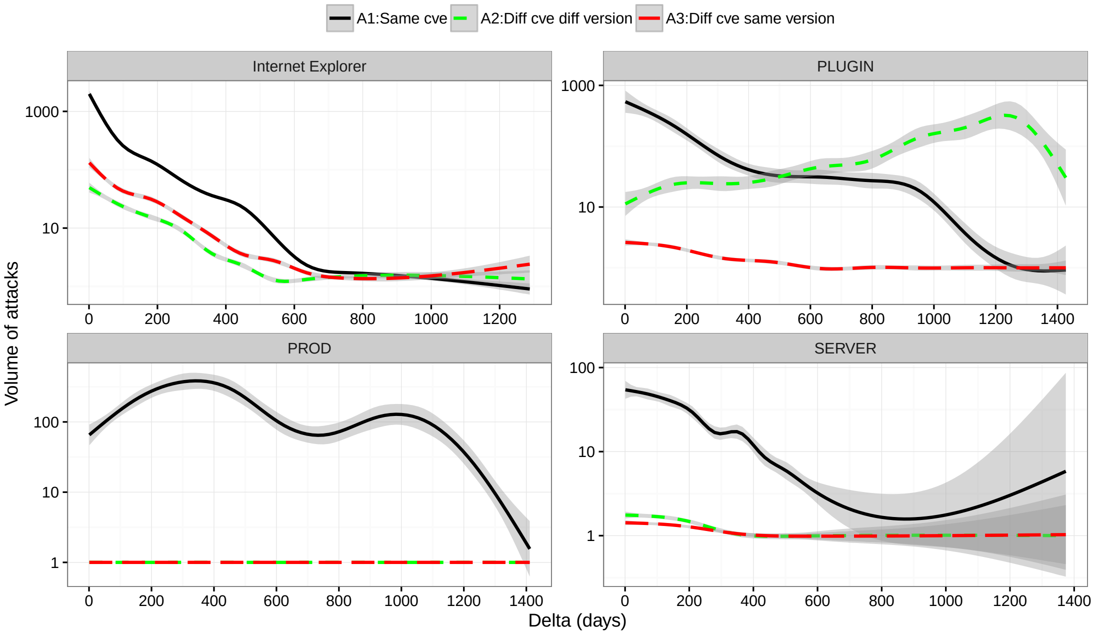

Figure 4: Loess regression of volume of attacks in time.

Note: Volume of received attacks as a function of time for the three types of attack. A_1 is represented by a solid black line; A_2 by a long-dashed red line; A_3 by a dashed green line. The grey areas represent 95% confidence intervals. For Internet Explorer vulnerabilities the maximum $\mathcal{T}$ between two attacks is 1288 days; for SERVER is 1374 days; PROD 1411; PLUGIN 1428. This can be determined by the timing of first appearance of the attack in the WINE database.

category. As expected, A_1 dominates in all software types. The predicted order is valid for PLUGIN and PROD. For PROD software we find no attacks against new vulnerabilities for different software versions, therefore $A_2 = A_3 = 0$. This may be an effect of the typically low update rate of this type of software and relatively short timeframe considered in our dataset (3 years), or of a scarce attacker interest in this software type. Results for SERVER are mixed as discussed above: the difference between A_2 and A_3 is very narrow and A_3 is occasionally higher than A_2. Since oscillations occur within the confidence intervals they might be due to chance as well.

Internet Explorer. is an interesting case in itself. Here, contrary to our prediction, A_3 is higher than A_2. By further investigating the data, we find that the reversed trend is explained by one single outlier pair: $\texttt{CVE}_1$ =CVE-2010-0806 and $\texttt{CVE}_2$ =CVE-2009-3672. These vulnerabilities affect Internet Explorer version 7 and have been disclosed 98 days apart, within our 120 days threshold. More interestingly, they are very similar: they both affect a memory corruption bug in Internet Explorer 7 that allows for an heap-spray attack resulting in arbitrary code execution. Two observations are particularly interesting to make:

1. Heap spray attacks are unreliable attacks that may result in a significant drop in exploitation success. This is reflected in the `Access Complexity:Medium` assessment assigned to both vulnerabilities by the CVSS v2 framework. In our model, this would reflected in a lower return $r(t, N_V(t), V)$ for the attacker, as the unreliable exploit may yield control of fewer machines than those that are vulnerable.
2. The exploitation code found on Exploit-DB[9] is essentially the same for these two vulnerabilities. The code for $\texttt{CVE}_2$ is effectively a rearrangement of the code for $\texttt{CVE}_1$, with different variable names. In our model, this would indicate that the cost $C(v|V) \approx 0$ to build an exploit for the second vulnerability is negligible, as most of the exploitation code can be re-used from $\texttt{CVE}_1$.

Hence, this vulnerability pair is only an apparent exception: the very nature of the second exploit for Internet Explorer 7 is coherent with our model and in line with Hyp. 1 and Hyp. 2. Removing the pair from the data confirms the order of attack scenarios identified in Table 7.

We now check how the trends of attacks against a software change with time. Hyp. 3 states that the exploitation of the same vulnerability persists in time and decreases slowly at a pace depending on users' update behaviour. This hypothesis offers an alternative behavior with respect to other models in literature where new exploits arrive very quickly after the date of disclosure, and attacks increase following a steep curve as discussed by Arora et al. (2004).

4.1. An Econometric Model of the Engineering of Exploits

We can use Proposition 2 to identify a number of additional hypothesis that are useful to formulate the regression equation. At first we notice that $T^\star = O(\log(\theta_v - \theta_V)N)$. Therefore we have a first identification relation between the empirical variable $\mathcal{U}$ (corresponding to N) and the empirical variable $\mathcal{T}$ (whose correspondence to $T^\star$ is outlined later in this section).

Hypothesis 4. *There is a log-linear relation between the number of attacked systems $\mathcal{U}$ and the delay $\mathcal{T}$.*

Since $\partial T^\star / \partial((\theta_v - \theta_V)N) < 0$ a larger number of attacked systems $\mathcal{U}$ on *different* versions ($\theta_v \neq \theta_V$) would imply a lower delay $\mathcal{T}$ (as there is an attractive number of new systems that guarantee the profitability of new attacks). In contrast, the baseline rate of attacks impacts negatively the optimal time $\mathcal{T}$ as $\partial T^\star / \partial(\theta_V N) > 0$ since a larger pool of vulnerable machines makes it more profitable to continue with existing attacks (as per Hyp. 1).

Hypothesis 5. *The possibility of launching a large number of attacks against systems for which an exploit already exists lengthens the time for weaponizing a vulnerability ($\mathcal{N} \cdot (\texttt{Ver}_0 = \texttt{Ver}_v) \uparrow \implies \mathcal{T} \uparrow$), whereas an increase in potential attacks on different systems is an incentive towards a shorter weaponization cycle ($\mathcal{N} \cdot (\texttt{Ver}_0 \neq \texttt{Ver}_v) \uparrow \implies \mathcal{T} \downarrow$).*

[9] See Exploit-DB (`http://www.exploit-db.com`, last accessed January 15, 2017.), which is a public dataset for vulnerability proof-of-concept exploits. $\texttt{CVE}_1$ corresponds to exploit 16547 and $\texttt{CVE}_2$ corresponds to exploit 11683.

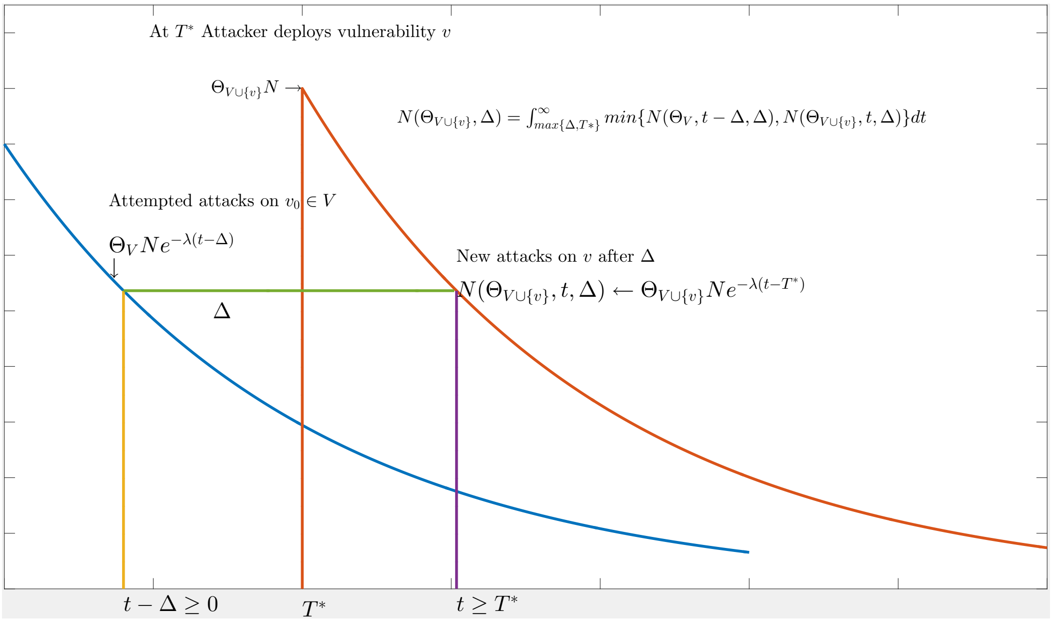
At T^* Attacker deploys vulnerability v
$\Theta_{V\cup\{v\}}N \rightarrow$
$N(\Theta_{V\cup\{v\}},\Delta) = \int_{max\{\Delta,T*\}}^{\infty} min\{N(\Theta_V, t-\Delta,\Delta), N(\Theta_{V\cup\{v\}},t,\Delta)\}dt$
Attempted attacks on $v_0 \in V$
$\Theta_V N e^{-\lambda(t-\Delta)}$ $\downarrow$
New attacks on v after Δ
$N(\Theta_{V\cup\{v\}},t,\Delta) \leftarrow \Theta_{V\cup\{v\}}Ne^{-\lambda(t-T^*)}$
Δ
$t-\Delta \geq 0$
T^*
$t \geq T^*$

Figure 5: Computing the delay ($\mathcal{T}$) against different vulnerabilities.

Note: Change in the number of attacked systems for two attacks against different systems $\Delta = \mathcal{T}$ days apart. The first attack happens at $t - \mathcal{T} \geq 0$ and the number of attacked systems $\mathcal{U}(\Theta_V \cup \{v\}, t, \mathcal{T})$ is derived from Eq. (1) as $\Theta_V N e^{-\lambda(t-\mathcal{T})}$. The number of systems attacked by the new exploit introduced at $T^\star$ is derived as $\mathcal{U}(\Theta_{V\cup\{v\}}, t, T^\star) = N\Theta_{V\cup\{v\}}e^{-\lambda(t-T^\star)}dt$.

When considering the effects of costs, we observe that, as $\partial T^\star/\partial C(v|V) > 0$, the presence of a vulnerability with a low attack complexity implies $dC(v|V) < 0$, and therefore reflects a drop in the delay $\mathcal{T}$ between the two attacks. We have already discussed this possibility as Hypothesis 2.

As for revenues, it is $\partial T^\star/\partial r < 0$ so that an higher expected profit would imply a shorter time to weaponization. However, we cannot exactly capture the latter condition since in our model the actual revenue r is stationary and depends only on the captured machine rather than the individual type of exploit. A possible proxy is available through the `Imp` variable, but it only shows the level of *technical* compromise that is possible to achieve. Unfortunately, such information might not correspond to the actual revenue that can be extracted by the attacker. For example, vulnerabilities that only compromise the availability of a system are scored low according to the CVSS standard. However, for an hacker offering "booter services" to on- line gamers (i.e. DDoS targeted attack against competitors) these vulnerabilities are the only interesting source of revenues Hutchings and Clayton (2016).

However `Imp` can also be seen as a potential conditional factor to boost the attractiveness of a vulnerability as the additional costs of introducing an exploit might be justified by the increased capability to produce more havoc.

Hypothesis 6. *Vulnerabilities with higher impact increase revenue and therefore decrease number of attacks ($\mathtt{Imp}_{\mathtt{CVE}_2} > \mathtt{Imp}_{\mathtt{CVE}_1} \implies \mathcal{U}\downarrow$).*

As the time of introduction of an exploit $T^\star$ can not be directly measured from our dataset, we use $\mathcal{T}$ (i.e. the time in between two consequent attacks) as a proxy for the same variable. Figure 5 reports a pictorial representation of the transformation. Each curve represents the decay in time of number of attacks against two different vulnerabilities. The first attack (blue line) is introduced at $t = 0$, and the second (red line) at $t = T^\star$.

The number of received attacks is described by the area below the curve within a certain interval. Let $\mathcal{U}(\Theta_V \cup \{v\}, t, \mathcal{T})$ represent the number of systems that receive two attacks $\mathcal{T}$ days apart, at times $t - \mathcal{T}$ and t respectively. Depending on the relative position of $t - \mathcal{T}$ with respect to $T^\star$, the interval within which reside the measured attacks on the pair of vulnerabilities will be $\int_{\max(T^\star,\mathcal{T})}^{\infty} \mathcal{U}(\cdot)dt$. Setting the number of attacks at time $t - \mathcal{T}$ as $\mathcal{U}(\theta_v, t - \mathcal{T}) = N\theta_v e^{-\lambda(t-\mathcal{T})}$ and the attacks received on the second vulnerability at time t as $\mathcal{U}(\theta_{V\cup\{v\}}, t) = N\theta_{V\cup\{v\}} e^{-\lambda(t-T^\star)}$, we obtain

$$\mathcal{U}(\theta_{V\cup\{v\}}, t, \mathcal{T}) = \min\left(N\theta_v e^{\lambda\mathcal{T})}, N\theta_{V\cup\{v\}} e^{\lambda T^\star}\right) \int_{\max(\mathcal{T},T^\star)}^{\infty} e^{-\lambda t} dt \tag{9}$$

Solving for the two cases $T^\star > \mathcal{T}$ and $T^\star < \mathcal{T}$, we formulate the following claim:

Claim 1.

$$\log\mathcal{U}(\theta_{V\cup\{v\}}, t, \mathcal{T}) = \begin{cases} \log\frac{N}{\lambda} - \lambda T^\star + \lambda\mathcal{T} + \log\theta_v & \textit{if } T^\star > \mathcal{T} \\ \log\frac{N}{\lambda} + \lambda T^\star - \lambda\mathcal{T} + \log\theta_{V\cup\{v\}} & \textit{if } T^\star < \mathcal{T} \end{cases} \tag{10}$$

The sign of the coefficient for $\mathcal{T}$ oscillates from positive to negative as $\mathcal{T}$ increases.

Proof of Claim 1 and its empirical evaluation are given in Appendix 6.4. □

Reviewing Figure 1, our data suggests that $\mathcal{T}$ is on average more than 100 days with respect to $T^\star$. Therefore we have:

$$\log\mathcal{U} = -\lambda\mathcal{T} + \lambda T^\star + \log\frac{N}{\lambda} + \log\theta_{V\cup\{v\}}$$

Substituting $T^\star$ from Eq. (8), the number of expected attacked systems after $\mathcal{T}$ days is:

$$\log\mathcal{U} = -\lambda\mathcal{T} + \lambda\left[\frac{1}{\delta}\log\left(\frac{C(v|V)}{r} - \frac{\delta}{\lambda+\delta}(\theta_{V\cup\{v\}} - \theta_V)N\right)\right] + \log\frac{N}{\lambda} + \log\theta_{V\cup\{v\}} \tag{11}$$

Our regression model tests the hypotheses above by reflecting the formulation provided in Eq. (11). $\mathcal{T}$ can be measured directly in our dataset; the cost of development of an exploits $C(v|V)$ can be estimated by the proxy variables $\texttt{Compl}_{\texttt{CVE}_2}$, as the complexity associated with exploit development requires additional engineering effort (and is thus related to an increase in development effort) CVSS-SIG (2015). We can not directly measure the revenue r and the number of systems N affected by the vulnerability, but we can estimate the effect of an attack on a population of users by measuring the impact (`Imp`) of that vulnerability on the system: higher impact vulnerabilities (i.e. ($\texttt{Imp}_{CVE2} > \texttt{Imp}_{CVE1}$) allow the attacker to control a *higher fraction* of the vulnerable system, and therefore extract higher revenue r from the attack. Similarly, the introduction of an attack with a higher impact can approximate the difference in attack penetration $(\theta_{V\cup\{v\}} - \theta_V)N$ for the new set of exploits as it allows the attacker for a higher degree of control on the affected systems. Finally, high impact vulnerabilities ($\texttt{Imp}_{CVE2,H}$), for example allowing remote execution of arbitrary code on the victim system, leave the $\Theta_{V\cup\{v\}}N$ systems under complete control of the attacker; in contrast, a low impact vulnerability, for example causing a denial of service, would allow for only a temporary effect on the machine and therefore a lower degree of control. In Table 8 we report the sample correlation matrix for the variables included in the regression system we will use to parameterize the model, from an econometric standpoint, the highest pairwise correlations with $\mathcal{T}_i$ are `Frq` MEDIUM and `Pk` HIGH, however these have correlations of less than 20%, as such the standard issues on rank and collinearity are not present.

Table 8: Correlation Matrix of All Variables Included in the Model.

Model variable	1.	2.	3.	4.	5.	6.	7.	8.	9.	10.	11.
1. $\mathcal{T}$	1										
2. $\texttt{Compl}_{CVE2L}$	-0.130	1									
3. $\texttt{Imp}_{CVE2H}$	0.058	-0.237	1								
4. $\texttt{Imp}_{CVE2} > \texttt{Imp}_{CVE1}$	0.092	-0.097	0.055	1							
5. `Geo` North. Am.	0.020	0.146	-0.040	-0.035	1						
6. `Geo` Western Eu.	0.022	-0.026	0.011	-0.051	-0.410	1					
7. `Hst` EVOLVE	-0.084	0.024	-0.065	0.022	-0.057	0.015	1				
8. `Hst` UPGRADE	0.054	0.004	0.030	-0.017	0.084	-0.042	-0.656	1			
9. `Frq` HIGH	0.104	-0.050	0.052	0.107	0.087	-0.183	-0.225	-0.024	1		
10. `Frq` MEDIUM	0.136	0.077	-0.103	0.127	0.014	0.096	-0.087	0.186	-0.369	1	
11. `Pk` HIGH	0.166	-0.008	0.045	0.049	0.020	0.033	-0.288	0.034	0.678	0.052	1
12. `Pk` MEDIUM	-0.087	0.031	-0.054	-0.004	-0.023	-0.008	0.197	0.004	-0.549	0.101	-0.814

Table 9: Summary of predictions derived from the model.

Model variable	Regressor	Expectation	Hyp.	Rationale
$\mathcal{T}$	$\mathcal{T}$	$\beta_1 < 0$	Hyp. 3, Hyp. 4, Hyp. 5	Shorter exploitation times are associated with more vulnerable systems, hence $\mathcal{T}\uparrow \implies \mathcal{U}\downarrow$.
$C(V\|v)$	$\texttt{Compl}_{CVE2,L}$	$\beta_2 < 0$	Hyp. 1, Hyp. 5, Hyp. 2	The introduction of a new reliable, low-complexity exploit minimizes implementation costs, thus $C \downarrow \implies \mathcal{U}\downarrow$.
$\theta_{V\cup\{v\}}$	$\texttt{Imp}_{CVE2,H}$	$\beta_3 > 0$	Hyp. 6, Hyp. 5	High impact vulnerabilities allow the attacker for a complete control of the attacked systems, hence $\theta_{V\cup\{v\}} \uparrow \implies \mathcal{U}\uparrow$.
r, $(\theta_{V\cup\{v\}} - \theta_V)$	$\texttt{Imp}_{CVE2} > \texttt{Imp}_{CVE1}$	$\beta_4 < 0$	Hyp. 6	Selecting a higher impact exploit for a new vulnerability increases the expected revenue and increases the fraction of newly controlled systems with respect to the old vulnerability. $r \uparrow \implies \mathcal{U}\downarrow$ and $(\theta_{V\cup\{v\}} - \theta_V)\uparrow \implies \mathcal{U}\downarrow$.

To test our hypotheses, we set three equations to evaluate the effect of our regressors on the dependent variable. The formulation is derived from prime principles from Eq. (11) as discussed above. Our equations are:

$$\begin{aligned}\text{Model 1:}\quad \log(\mathcal{U}_i) &= \beta_0 + \beta_1\mathcal{T}_i + \epsilon_i && (12)\\ \text{Model 2:}\quad \log(\mathcal{U}_i) &= \dots + \beta_2\texttt{Compl}_{i,CVE2,L} + \epsilon_i && (13)\\ \text{Model 3:}\quad \log(\mathcal{U}_i) &= \dots + \beta_3\texttt{Imp}_{i,CVE2,H} + \beta_4(\texttt{Imp}_{i,CVE2} > \texttt{Imp}_{i,CVE1})\epsilon_i && (14)\end{aligned}$$

Where i indexes the pair of attacks received by each machine after $\mathcal{T}$ days, $\texttt{Compl}_{i,CVE2,L}$ indicates that $\texttt{CVE}_2$ has a low complexity, and $\texttt{Imp}_{i,CVE2,H}$ indicates that $\texttt{CVE}_2$ has a *High* (≥ 7) impact. Both classifications for `Compl` and `Imp` are reported by the CVSS standard specification. The correlation matrix of the model variables is reported in Table 8.

The mapping of each term with our hypotheses and the predicted values of the regressors are described in the Table 9. Further, we add the vector of controls Z to the regression to account for exogenous factors that may confound the observation, as discussed in Section 3.2:

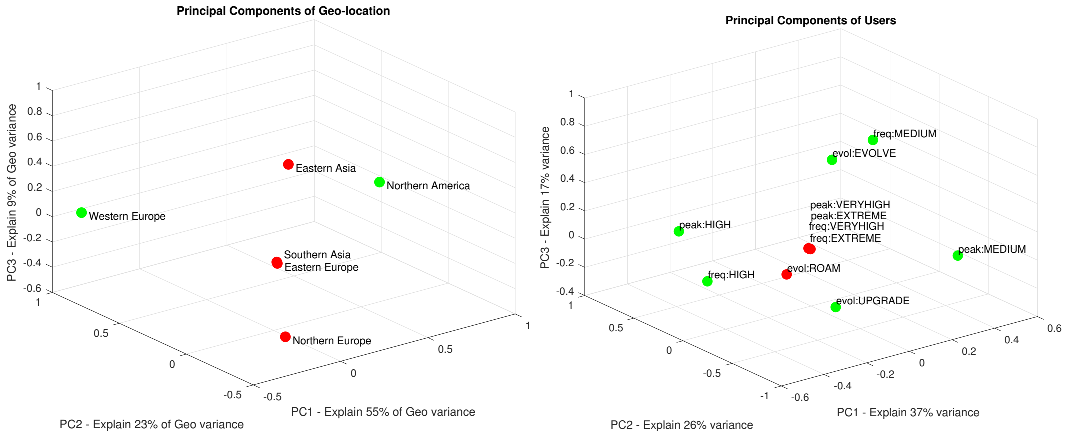

Figure 6: Principal component analysis of location and user factors.
Each dot corresponds to the loading values for the geographical location and user specific factors denoted `Geo`, `Hst`, `Frq`, and `Pk` factors. The axes report the loadings of the components for the eigenvalues that explain the most variance. The farthest points from the graph origin (in green) are factor levels with the highest level of independence.

`Geo`, `Hst`, `Frq`, `Pk`. Z is defined as

$$Z_i^{all} = \sum_{d \in \texttt{Geo}} \alpha_d(\texttt{Geo}_i = d) + \sum_{d \in \texttt{Hst}} \alpha_d(\texttt{Hst}_i = d) + \sum_{d \in \texttt{Frq}} \alpha_d(\texttt{Frq}_i = d) + \sum_{d \in \texttt{Pk}} \alpha_d(\texttt{Pk}_i = d) \quad (15)$$

Given the model, it is reasonable to expect the variance in attacked systems to be related to the level of some independent variables. For example, the variance of the dependent variable $\mathcal{U}$ may depend on the value of $\mathcal{T}$. To address possible heteroskedasticity of the data, we employ a robust regression next to a regular OLS and compare the results. To evaluate the independence between the control factor levels over the respective contingency table we employ a standard Principal Component Analysis. Figure 6 shows a plot of the analysis for loadings of the principal components in Table 2.

We select all eigenvalues that explain at least 10% of the variance: `Geo` has only three eigenvalues above the threshold, whereas `Hst`, `Frq`, and `Pk` have four eigenvalues above the threshold. Altogether they explain 87% of the variance for geo-location and 80% for user characteristics. The corresponding components of the eigenvectors (corresponding to the loadings of the controls) are identified and we select the control values that have the greatest distance from the origin in the eigenvector space (at least 10% again as a sum of squares). This guarantees that we select the controls with the highest degree of independence. This results in eight selected controls, identified in green in Figure 6.

Our regression results are reported in Table 10. We utilize two estimators as we have little information on the error structure of the regression model. First is a simple OLS estimator with Huber-White standard errors and second is a Robust fit model that utilizes a WLS type estimator with iterative re-weighting and again we implement the sandwich form standard error from the WLS iterations. The weighting function for the iterative re-weighting is a bisquare function, experimentation with spectral and Andrews type weightings suggest the regressions are insensitive to kernel and tuning function. This indicates that the OLS estimates are very reliable for this purpose. For the robust fit we compute a McFadden adjusted pseudo-R^2, which sets the numerator as the log likelihood function at the estimate and the denominator as the log likelihood of just the intercept alone. Note that it is not appropriate to compare directly the pseudo-R^2 and the R^2 from the OLS estimates, which suggests that the model captures roughly 10% of the variation in numbers of attacked machines, as opposed to explaining 35% of the model likelihood for the pseudo-R^2.

The set of OLS and Robust regressions returns very similar estimations. The introduction of the controls only change the sign of β_1 from positive to negative for Model 1. This may indicate that the type of user is a significant factor in determining the number of delivered attacks, which is consistent with previous findings Nappa et al. (2015). Interestingly, the factor that introduces the highest change in the estimated coefficient β_1 for $\mathcal{T}$ is `Compl` (Model 2), whereas its estimate remains essentially unchanged in Model 3. This may indicate that the cost of introduction of an exploit has a direct impact on the time of delivery of the exploit. The coefficients for all other regressors are consistent across models, and their magnitude changes only slightly with the introduction of the controls. This should be expected: user characteristics should not influence the characteristics of the vulnerabilities present on the system. Hence, the distribution of attacks in the wild seems to be prevalently dependent on system characteristics and independent of user type. The signs of the coefficients for the `Imp` variables suggest that both the impact of a vulnerability and the relation of the impact of the new vulnerability w.r.t. previous ones have an effect on the number of attacked systems. Interestingly, a *high* impact encourages the deployment of attacks and increases the number of attacked systems, whereas the introduction of a *higher* impact vulnerability requires the infection of a smaller number of systems as revenues extracted from each machine increase. This indicates that when introducing a new exploit, the attacker will preferably choose one that grants a higher control over the population of users ($\theta_{V \cup \{v\}} > \theta_V$) and use it against a large number of system. This goes in the same direction of recent findings that suggest that vulnerability severity alone is not a good predictor for exploitation in the wild Allodi and Massacci (2014), Bozorgi et al. (2010), and that other factors such as software popularity or market share may play a role Nayak et al. (2014).

5. Discussion, Conclusions and Implications

This paper implements a model of the *Work-Averse Attacker* as a new conceptual framing to understand cyber threats. Our model presumes that an attacker is a resource-limited actor with fixed costs that has to choose which vulnerabilities to exploit to attack the 'mass of Internet systems'. Work aversion simply means that effort for the attacker is costly (in terms of cognition and opportunity costs), hence a trade-off exists between effort exerted on new attacking technologies and the anticipated reward schedule from these technologies. As technology combinations mature, their revenue streams are presumed dwindle.

In this framework, an utility-maximizing attacker will drive exploit production according to their expectations that the newly engineered attack will increase net profits from attacks against the general population of internet users. As systems in the wild get patched unevenly and often slowly in time (Nappa et al. 2015), we model the production of new vulnerability exploits following Stokey's 'economy of inaction' logic, whereby 'doing nothing' before a certain (time) threshold is the best strategy. From the model a cost constraint driving the attacker's exploit selection strategy naturally emerges. In particular, we find theoretical and empirical evidence for the following:

1. An attacker massively deploys only one exploit per software version. The only exception we find is for Internet Explorer; the exception is characterised by a very low cost to create an additional exploit, where it is sufficient to essentially copy and paste code from the old exploit, with only few modifications, to obtain the new one. This finding supports Hyp. 1.

Table 10: Ordinary Least Squares and Robust Regression Results

Dependent Variable: natural logarithm of the number of attacked machines $\log(\mathcal{U}_i)$

	Model 1				Model 2				Model 3			
	OLS		Robust		OLS		Robust		OLS		Robust	
		$Z1:Z8$		$Z1:Z8$		$Z1:Z8$		$Z1:Z8$		$Z1:Z8$		$Z1:Z8$
c	0.927	0.006	0.731	0.096	1.065	0.122	0.845	0.171	0.933	-0.106	0.783	0.039
	(0.001)	(0.003)	(0.001)	(0.003)	(0.001)	(0.003)	(0.001)	(0.003)	(0.004)	(0.005)	(0.003)	(0.004)
$\mathcal{T}$	0.018	-0.051	0.012	-0.044	-0.006	-0.092	-0.003	-0.071	-0.005	-0.091	-0.004	-0.071
	(0.001)	(0.001)	(0.001)	(0.001)	(0.001)	(0.001)	(0.001)	(0.001)	(0.001)	(0.001)	(0.001)	(0.001)
$\texttt{Compl}_{CVE2L}$					-0.326	-0.479	-0.228	-0.324	-0.313	-0.464	-0.22	-0.314
					(0.002)	(0.002)	(0.001)	(0.001)	(0.002)	(0.002)	(0.001)	(0.001)
$\texttt{Imp}_{CVE2H}$									0.144	0.236	0.063	0.131
									(0.003)	(0.003)	(0.003)	(0.003)
$\texttt{Imp}_{CVE2} > \texttt{Imp}_{CVE1}$									-0.088	-0.209	0.012	-0.087
									(0.003)	(0.003)	(0.002)	(0.002)
Z1: `Geo` North. Amer.		0.604		0.37		0.679		0.422		0.671		0.419
		(0.002)		(0.001)		(0.002)		(0.001)		(0.002)		(0.001)
Z2: `Geo` West. Eu.		0.155		0.105		0.17		0.116		0.163		0.114
		(0.002)		(0.002)		(0.002)		(0.002)		(0.002)		(0.002)
Z3: `Hst` EVOLVE		0.191		0.129		0.208		0.141		0.223		0.149
		(0.002)		(0.002)		(0.002)		(0.002)		(0.002)		(0.002)
Z4: `Hst` UPGRADE		0.112		0.072		0.116		0.076		0.113		0.075
		(0.002)		(0.002)		(0.002)		(0.002)		(0.002)		(0.002)
Z5: `Frq` HIGH		0.24		0.147		0.212		0.127		0.279		0.157
		(0.003)		(0.003)		(0.003)		(0.003)		(0.003)		(0.003)
Z6: `Frq` MEDIUM		0.328		0.227		0.358		0.246		0.41		0.271
		(0.002)		(0.002)		(0.002)		(0.002)		(0.002)		(0.002)
Z7: `Pk` HIGH		0.513		0.442		0.567		0.49		0.531		0.477
		(0.004)		(0.003)		(0.004)		(0.003)		(0.004)		(0.003)
Z8: `Pk` MEDIUM		0.379		0.274		0.412		0.299		0.411		0.301
		(0.003)		(0.002)		(0.003)		(0.002)		(0.003)		(0.002)
$Pseudo\ R^2$	-	-	0.326	0.341	-	-	0.331	0.347	-	-	0.331	0.347
R^2	0.00	0.093	-	-	0.016	0.126	-	-	0.017	0.13	-	-
F	348.661	26,551.467	-	-	18,548.248	33,422.784	-	-	9,989.879	28,915.597	-	-
Obs.	2,324,500	2,324,500	2,324,500	2,324,500	2,324,500	2,324,500	2,324,500	2,324,500	2,324,500	2,324,500	2,324,500	2,324,500

$$
\begin{aligned}
\text{Model 1:}\quad \log(\mathcal{U}_i) &= \beta_0 + \beta_1 \mathcal{T}_i + \epsilon_i \\
\text{Model 2:}\quad \log(\mathcal{U}_i) &= \beta_0 + \beta_1 \mathcal{T}_i + \beta_2 \texttt{Compl}_{i,CVE2,L} + \epsilon_i \\
\text{Model 3:}\quad \log(\mathcal{U}_i) &= \beta_0 + \beta_1 \mathcal{T}_i + \beta_2 \texttt{Compl}_{i,CVE2,L} + \beta_3 \texttt{Imp}_{i,CVE2,H} + \beta_4 \texttt{Imp}_{i,CVE2} > \texttt{Imp}_{i,CVE1} \epsilon_i
\end{aligned}
$$

Notes: The three model equations reflect the definition of the expected (log) number of affected machines after an interval $\mathcal{T}$. The regression model formulation is derived from prime principle from Eq. 11. The expected coefficient signs are given in Table 9. For each model we run four sets of regressions. OLS and Robust regressions are provided to addresses heteroscedasticity in the data. R^2 and $F - statistics$ are reported for the OLS estimations. Note that the pseudo-R^2 are computed for the robust regressions, using the McFadden adjusted approach $R^2 = 1 - (\log(LL_{full}) - K)/\log(LL_{int})$, where $\log(LL_{full})$is the log likelihood for the full model minus the number of slope parameters K versus the log likelihood of the intercept alone and should not be compared directly to the OLS R^2. Coefficient estimations of the two sets of regressions are consistent. All regressions are run with and without the set of controls defined by the PCA analysis pictorially reported in Fig. 6.

All coefficient signs for the three models reflect the work-averse attacker model predictions, with the only exception of the estimation for $\mathcal{T}$ with no controls for which the prediction for β_1 is inverted. This may indicate that user characteristics are relevant factors for the arrival time of exploits when other factors related to the system are not accounted for. The introduction of `Compl` in Model 2 significantly changes the estimate for β_1, whereas `Imp` in Model 3 leaves the estimates for `Compl` and $\mathcal{T}$ unchanged. High `Imp` vulnerabilities tend to increase volume of attacks. We report only standard errors without starring p-values as all coefficients are significant due to the number of observations in the dataset. All standard errors are estimated using the Huber-White approach.

2. Low complexity vulnerabilities for which a reliable exploit can be easily engineered lower the production costs and favor the deployment of the exploit. This finding supports Hyp. 2.
3. The attacker deploys new exploits relatively slowly over time, driven by a slowly decreasing instantaneous profit function; empirically, we find that attacks 1000 days apart are still driven by the same exploits in about 20% of the cases, and that the effect of the passage of time in between attacks ($\mathcal{T}$) on the number of affected system is indeed negative and very small given the current patching rate. This finding supports Hyp. 3 and Hyp. 5.
4. The presence of a high impact vulnerability increases the incidence of exploitation in the wild. Similarly, gaining a higher control over the attacked systems heightens the attacker's revenue and decreases the number of systems that need to be infected to balance development costs. This supports Hyp. 6.

The above findings suggest that risk associated with software vulnerabilities can not be solely measured in terms of technical severity, and that other, measurable factors may be included in the assessment, as previously suggested by some authors (see for example Bozorgi et al. 2010, Houmb et al. 2010, Allodi and Massacci 2014). While characteristics of the attacker and of the user, such as skill or competence, in system security may remain hard to measure, this paper shows how measurable environmental indexes may be used to estimate the economic incentives that exists for the attacker (as suggested, for example, by Anderson 2008).

5.1. Implications For Theory

The modeling of cyber attacks and cyber risk has traditionally been centered on the vulnerabilities present on systems and their technical severity. In recent years, this perspective has been largely questioned as the typically limited attacker resources became empirically apparent (Grier et al. 2012), and advances in security modeling started distinguishing between opportunistic (i.e. untargeted) and deliberate (i.e. targeted) attacks (see Ransbotham and Mitra 2009, for a discussion on this). However, current studies modeling cybersecurity events typically think of the attacker as employing an undefined 'attack generation function', which too easily collapses to an 'all-powerful' attacker that may generate an attack for whichever vulnerability exists.

Little theoretical discourse has been developed around the identification of models of vulnerability exploit production. Most assessments on whether the attacker will be successful are still produced by means of 'expert estimates' (Wang et al. 2008) or 'technical measures of vulnerabilities' (Naaliel et al. 2014) that, however, are known *not* to correlate with attacker choices as shown by Bozorgi et al. (2010). Researchers noted that "some vulnerabilities are different than others" (Nayak et al. 2014), but a rationale enabling this distinction between vulnerabilities is yet to be fully discussed.

In contrast to the classic 'all powerful attacker' model (Dolev and Yao 1983b) that can and will exploit potentially any vulnerability, this paper develops the thesis that the utility-maximizing attacker will generally 'avoid to work' until the perceived utility of the deployment of a new attack becomes positive w.r.t. to expectations derived from the previous attack at time $T^\star$. This economic perspective has been previously employed in game-theoretic approaches (Manshaei et al. 2013), but it typically considers two actors (the organization - namely the defender and the attacker) that react to each other's strategies. While this is significant for the case of targeted attacks, where the attacker can observe part or all of the organization's defenses and the defender chooses a strategy to decrease its attack surface (Howard et al. 2005), it is unclear how this maps to the case of the 'general attacker' that deploys attacks against the vast Internet population. As the average Internet user has little to no knowledge of computer and network security (Wash 2010), they are unlikely to deploy defensive strategies and to react strategically to new attacks. Hence, in this case, the attacker operates in an environment where relevant variables are not entirely exogenously manipulated by an intelligent adversary (from the attacker's perspective, the defender). The strategy of the mass of defenders may only be statistically determined.

While untargeted attacks exploiting software vulnerabilities make up for a significant fraction of all attacks recorded (Provos et al. 2008, Baker et al. 2012), this particular setting is largely unexplored in literature. As there is virtually no 'purely reactive strategy' to the decision of the attacker, part of the debate on regulatory intervention may focus on how to devise regulatory models that, on average, will increase the development costs for the attacker. Even targeted attacks require fixed costs and it is unclear whether such attacks could be captured by variation in the work aversion (or by making reward a function of costs such as reconnaissance) (Verizon 2011, Bilge and Dumitras 2012).

5.2. Implications For Information Security Management and Policy

Our findings suggest that the rationale behind vulnerability exploitation could be leveraged by defenders to deploy more cost-effective security countermeasures. For example, it is well known that software updates correspond to an increased risk of service disruption (e.g. for incompatibility problems or updated/deprecated libraries). However, if most of the risk for a particular software version comes from a specific vulnerability, than countermeasures other than patching may be more cost-effective. For example, maintaining network IDS signatures may be in this case a better option than updating the software, because one IDS signature could get rid of the great majority of risk that characterizes that system while a software patch may 'overdo it' by fixing more vulnerabilities than necessary.

Further, this view on the attacker has repercussions on the impact of vulnerability disclosure in terms of security of the user or organization. Work in this direction explored both the economic incentives for sharing information security (Gal-Or and Ghose 2005, Ransbotham et al. 2012) and the impact of vulnerability disclosure on attacks and firm value (Arora et al. 2008, Telang and Wattal 2007). Some of this discussion resulted in recent open debates regarding policies for vulnerability disclosure, and the drafting of ISO standards to guide vulnerability communication to the vendor and to the public (e.g. ISO/IEC 29147:2014). For example, the United States Department of Commerce NTIA forum for vendors, industry players, and security researchers to discuss procedures and timings of the vulnerability disclosure process. [10] However, this discussion is not currently guided by a theoretical framework that can act as a supporting tool for the decision maker. For example, this may be applied to the case of vulnerability disclosure to evaluate or estimate the effect in terms of the effective increase in risk of attacks that follows the disclosure, extending previous work in this same direction by Mitra and Ransbotham (2015).

Further, a more precise and data-grounded understanding of the attacker poses a strategic advantage for the defender. For example, software diversification and code differentiation has already been proposed as a possible alternative to vulnerability mitigation (Chen et al. 2011, Homescu et al. 2013). By diversifying software the defender effectively decreases the number of systems the attacker can compromise with one exploit, effectively making the existence conditions for Eq. (8) harder to satisfy than by means of a patch release strategy

[10] The NTIA forum can be found at `https://www.ntia.doc.gov/other-publication/2016/multistakeholder-process-cybersecurity-vulnerabilities`, last accessed January 15, 2017.

employed by vendors. For example, software vendors may randomize the distribution of vulnerability patches to their users, to minimize the attacker's chances of matching their exploits with the vulnerabilities actually present on the system. A random distribution of patches would simply decrease the fraction of attackable systems regardless of the attacker's choice in which vulnerability to exploit. Moreover, diversifying defenses may be in fact less onerous than re-compiling code bases (when possible) or maintaining extremely diverse operational environments.

5.3. Limitations and Extensions

Our model describes an attacking adversary with costly production of malware. The model aims at explaining the attacker's preference in exploiting one vulnerability over another rather than casting a wide net immediately after a new vulnerability has been discovered. However, this preference can not be directly measured, but must be inferred through their real attack signatures from a record of attempted security incursions.

Records of attacks detected over a user's machine are necessarily conditioned over the user's proneness in receiving a particular attack. For example, a user may be inclined to open executable email attachments, but not in visiting suspicious websites. Thus, there may be a disassociation between the observed attacks and those engineered by the attacker. For our empirical dataset this limitation is mitigated by *WINE* reporting attack data on a very large representative sample of Internet users (Dumitras and Shou 2011). However, we also need to have additional conditioning variables to permit identification of the impact on various behavioral characteristics. Many of the additional characteristics of users that may influence the observed volume of attacks, such as educational level and culture which, are very difficult or close to impossible to gauge at the scale of data presented in this paper. As proxies to control for this effect we employ the `User Profile`, `Frequency`, `Peak` and geographic location variables, as these outline the user's proneness in receiving attacks. Further, geographic location may not only influence effects related to user culture, but also on attack diffusion.

Software versioning information is known to be unreliable at times with respect to vulnerability existence (Nguyen et al. 2015). Further, software versions can not be easily 'ordered' throughout software types, as different vendors adopt different naming schemes for software releases (Christey and Martin 2013, for an overview). We can not therefore order software versions over time easily. This is however irrelevant to our study as we are interested in measuring the sequences of newer attacks received by internet users, as opposed to measuring the existence of new exploits for subsequent software releases, our model predicts that the attacker will perform this information filtration dynamically as they view the rewards from their activities. A limitation of our empirical dataset is obviously the market penetration of Symantec, as of 2016 Symantec self reports that it is the largest security vendor[11] by market share in anti-virus and overall software security and hence has a broad coverage recording attacks on customers. However, third party verifiable measurement of these claims is difficult hence replication studies with different security vendors would be welcomed and given the simplicity of our regression specification easily implemented.

References

Allodi, L. (2015). The heavy tails of vulnerability exploitation. In *Proceedings of the 2015 Engineering Secure Software and Systems Conference (ESSoS'15).*

[11] Indeed, according to the 2016 market share Symatec has held this position for the last 15 years, see `https://www.symantec.com/en/uk/about/newsroom/analyst/reports`

Allodi, L., M. Corradin, and F. Massacci (2015). Then and now: On the maturity of the cybercrime markets the lesson that black-hat marketeers learned. *IEEE Transactions on Emerging Topics in Computing 4*(1), 35–46.

Allodi, L., V. Kotov, and F. Massacci (2013). Malwarelab: Experimentation with cybercrime attack tools. In *Proceedings of the 2013 6th Workshop on Cybersecurity Security and Test.*

Allodi, L. and F. Massacci (2014, August). Comparing vulnerability severity and exploits using case-control studies. *ACM Transaction on Information and System Security (TISSEC) 17*(1).

Anderson, R. (2008). Information security economics - and beyond. In *Proceedings of the 9th International Conference on Deontic Logic in Computer Science*, DEON '08, pp. 49–49.

Arora, A., R. Krishnan, A. Nandkumar, R. Telang, and Y. Yang (2004). Impact of vulnerability disclosure and patch availability-an empirical analysis. In *Proceedings of the 3rd Workshop on Economics and Information Security.*

Arora, A., R. Telang, and H. Xu (2008). Optimal policy for software vulnerability disclosure. *Management Science 54*(4), 642–656.

Asghari, H., M. Van Eeten, A. Arnbak, and N. Van Eijk (2013). Security economics in the https value chain. In *Twelfth Workshop on the Economics of Information Security (WEIS 2013), Washington, DC.*

August, T. and T. I. Tunca (2006). Network software security and user incentives. *Management Science 52*(11), 1703–1720.

August, T. and T. I. Tunca (2008). Let the pirates patch? an economic analysis of software security patch restrictions. *Information Systems Research 19*(1), 48–70.

August, T. and T. I. Tunca (2011). Who should be responsible for software security? a comparative analysis of liability policies in network environments. *Management Science 57*(5), 934–959.

Baker, W., M. Howard, A. Hutton, and C. D. Hylender (2012). 2012 data breach investigation report. Technical report, Verizon.

Baltazar, J. (2011). More traffic, more money: Koobface draws more blood. Technical report, TrendLabs.

Beattie, S., S. Arnold, C. Cowan, P. Wagle, C. Wright, and A. Shostack (2002). Timing the application of security patches for optimal uptime. In *LISA*, Volume 2, pp. 233–242.

Bilge, L. and T. Dumitras (2012). Before we knew it: an empirical study of zero-day attacks in the real world. In *Proceedings of the 19th ACM Conference on Computer and Communications Security (CCS'12)*, pp. 833–844. ACM.

Birge, J. R. (2010). The persistence and effectiveness of large-scale mathematical programming strategies: projection, outer linearization, and inner linearization. In *A Long View of Research and Practice in Operations Research and Management Science*, pp. 23–33. Springer.

Birge, J. R. and F. Louveaux (2011). *Introduction to stochastic programming.* Springer Science & Business Media.

Bozorgi, M., L. K. Saul, S. Savage, and G. M. Voelker (2010, July). Beyond heuristics: Learning to classify vulnerabilities and predict exploits. In *Proceedings of the 16th ACM International Conference on Knowledge Discovery and Data Mining.*

Brand, M., C. Valli, and A. Woodward (2010). Malware forensics: Discovery of the intent of deception. *The Journal of Digital Forensics, Security and Law: JDFSL 5*(4), 31.

Carlini, N. and D. Wagner (2014). Rop is still dangerous: Breaking modern defenses. In *23rd USENIX Security Symposium (USENIX Security 14)*, pp. 385–399.

Chen, P.-y., G. Kataria, and R. Krishnan (2011). Correlated failures, diversification, and information security risk management. *MIS Quarterly 35*(2), 397–422.

Christey, S. and B. Martin (2013, July). Buying into the bias: why vulnerability statistics suck. `https://www.blackhat.com/us-13/archives.html#Martin`.

CVSS-SIG (2015). Common vulnerability scoring system v3.0: Specification document. Technical report. `First.org`.

DeGroot, M. H. (2005). *Optimal statistical decisions*, Volume 82. John Wiley & Sons.

Dolev, D. and A. Yao (1983a). On the security of public key protocols. *IEEE Transactions on information theory 29*(2), 198–208.

Dolev, D. and A. Yao (1983b, mar). On the security of public key protocols. *IEEE Transactions on Information Theory 29*(2), 198 – 208.

Dumitras, T. and D. Shou (2011). 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*, pp. 89–96. ACM.

Erickson, J. (2008). *Hacking: the art of exploitation.* No Starch Press.

Finifter, M., D. Akhawe, and D. Wagner (2013). An empirical study of vulnerability rewards programs. In *Presented as part of the 22nd USENIX Security Symposium (USENIX Security 13)*, Washington, D.C., pp. 273–288. USENIX.

Franklin, J., V. Paxson, A. Perrig, and S. Savage (2007). An inquiry into the nature and causes of the wealth of internet miscreants. In *Proceedings of the 14th ACM Conference on Computer and Communications Security (CCS'07)*, pp. 375–388.

Frederick, S., G. Loewenstein, and T. O'donoghue (2002). Time discounting and time preference: A critical review. *Journal of economic literature 40*(2), 351–401.

Frei, S., D. Schatzmann, B. Plattner, and B. Trammell (2010). Modeling the security ecosystem - the dynamics of (in)security. In T. Moore, D. Pym, and C. Ioannidis (Eds.), *Economics of Information Security and Privacy*, pp. 79–106. Springer US.

Gal-Or, E. and A. Ghose (2005). The economic incentives for sharing security information. *Information Systems Research 16*(2), 186–208.

Gordon, L. A. and M. P. Loeb (2002). The economics of information security investment. *ACM Transactions on Information and System Security 5*(4), 438–457.

Grier, C., L. Ballard, J. Caballero, N. Chachra, C. J. Dietrich, K. Levchenko, P. Mavrommatis, D. McCoy, A. Nappa, A. Pitsillidis, N. Provos, M. Z. Rafique, M. A. Rajab, C. Rossow, K. Thomas, V. Paxson, S. Savage, and G. M. Voelker (2012). Manufacturing compromise: the emergence of exploit-as-a-service. In *Proceedings of the 19th ACM Conference on Computer and Communications Security (CCS'12)*, pp. 821–832. ACM.

Herley, C. (2013). When does targeting make sense for an attacker? *IEEE Security and Privacy 11*(2), 89–92.

Herley, C. and D. Florencio (2010). Nobody sells gold for the price of silver: Dishonesty, uncertainty and the underground economy. *Economics of Information Security and Privacy*.

Homescu, A., S. Neisius, P. Larsen, S. Brunthaler, and M. Franz (2013). Profile-guided automated software diversity. In *2013 IEEE/ACM International Symposium on Code Generation and Optimization (CGO)*, pp. 1–11. IEEE.

Houmb, S. H., V. N. Franqueira, and E. A. Engum (2010). Quantifying security risk level from cvss estimates of frequency and impact. *Journal of Systems and Software 83*(9), 1622–1634.

Howard, M., J. Pincus, and J. Wing (2005). Measuring relative attack surfaces. *Computer Security in the 21st Century*, 109–137.

Hutchings, A. and R. Clayton (2016). Exploring the provision of online booter services. *Deviant Behavior 37*(10), 1163–1178.

Jonsson, E. and T. Olovsson (1997). A quantitative model of the security intrusion process based on attacker behavior. *IEEE Transactions on Software Engineering 23*(4), 235–245.

Kanich, C., C. Kreibich, K. Levchenko, B. Enright, G. M. Voelker, V. Paxson, and S. Savage (2008). Spamalytics: an empirical analysis of spam marketing conversion. In *Proceedings of the 15th ACM Conference on Computer and Communications Security (CCS'08)*, CCS '08, pp. 3–14. ACM.

Karthik Kannan, R. T. (2005). Market for software vulnerabilities? think again. *51*(5), 726–740.

Kotov, V. and F. Massacci (2013). Anatomy of exploit kits. In J. Jürjens, B. Livshits, and R. Scandariato (Eds.), *Engineering Secure Software and Systems: 5th International Symposium, ESSoS 2013*, pp. 181–196. Springer Berlin Heidelberg.

Laffont, J.-J. and D. Martimort (2009). *The theory of incentives: the principal-agent model.* Princeton University Press.

Liu, P., W. Zang, and M. Yu (2005, February). Incentive-based modeling and inference of attacker intent, objectives, and strategies. *ACM Transactions on Information and System Security 8*(1), 78–118.

Manadhata, P. K. and J. M. Wing (2011). An attack surface metric. *IEEE Transactions on Software Engineering 37*, 371–386.

Manshaei, M. H., Q. Zhu, T. Alpcan, T. Bacşar, and J.-P. Hubaux (2013, July). Game theory meets network security and privacy. *ACM Comput. Surv. 45*(3), 25:1–25:39.

Mell, P., K. Scarfone, and S. Romanosky (2007). A complete guide to the common vulnerability scoring system version 2.0. Technical report, FIRST, Available at `http://www.first.org/`

cvss.

Mellado, D., E. Fernández-Medina, and M. Piattini (2010). A comparison of software design security metrics. In *Proceedings of the Fourth European Conference on Software Architecture: Companion Volume*, ECSA '10, New York, NY, USA, pp. 236–242. ACM.

Miller, C. (2007). The legitimate vulnerability market: Inside the secretive world of 0-day exploit sales. In *Proceedings of the 6th Workshop on Economics and Information Security.*

Mitra, S. and S. Ransbotham (2015). Information disclosure and the diffusion of information security attacks. *Information Systems Research 26*(3), 565–584.

Murdoch, S. J., S. Drimer, R. Anderson, and M. Bond (2010). Chip and pin is broken. In *2010 IEEE Symposium on Security and Privacy*, pp. 433–446. IEEE.

Naaliel, M., D. Joao, and M. Henrique (2014). Security benchmarks for web serving systems. In *Proceedings of the 25th IEEE International Symposium on Software Reliability Engineering (ISSRE'14).*

Nappa, A., R. Johnson, L. Bilge, J. Caballero, and T. Dumitras (2015). The attack of the clones: a study of the impact of shared code on vulnerability patching. In *2015 IEEE Symposium on Security and Privacy*, pp. 692–708. IEEE.

Nayak, K., D. Marino, P. Efstathopoulos, and T. Dumitraş (2014). Some vulnerabilities are different than others. In *Proceedings of the 17th International Symposium on Research in Attacks, Intrusions and Defenses*, pp. 426–446. Springer.

Nguyen, V. H., S. Dashevskyi, and F. Massacci (2015). An automatic method for assessing the versions affected by a vulnerability. *Empirical Software Engineering*, 1–30.

Nikiforakis, N., F. Maggi, G. Stringhini, M. Z. Rafique, W. Joosen, C. Kruegel, F. Piessens, G. Vigna, and S. Zanero (2014). Stranger danger: exploring the ecosystem of ad-based url shortening services. In *Proceedings of the 23rd international conference on World wide web*, pp. 51–62. ACM.

NIST (2015). National institute of standards and technology, national vulnerability database. http://nvd.nist.gov.

Ooi, K. W., S. H. Kim, Q.-H. Wang, and K.-L. Hui (2012). Do hackers seek variety? an empirical analysis of website defacements. In *Proceedings of the 33rd International Conference on Information Systems (ICIS 2012).* AIS.

PCI-DSS (2010). PCI. https://www.pcisecuritystandards.org/documents/pci_dss_v2.pdf.

Provos, N., P. Mavrommatis, M. A. Rajab, and F. Monrose (2008). All your iframes point to us. In *Proceedings of the 17th USENIX Security Symposium*, pp. 1–15.

Quinn, S. D., K. A. Scarfone, M. Barrett, and C. S. Johnson (2010). Sp 800-117. guide to adopting and using the security content automation protocol (scap) version 1.0. Technical report, NIST.

Ransbotham, S. and S. Mitra (2009). Choice and chance: A conceptual model of paths to information security compromise. *Information Systems Research 20*(1), 121–139.

Ransbotham, S., S. Mitra, and J. Ramsey (2012). Are markets for vulnerabilities effective? *MIS Quarterly 36*(1), 43.

Rao, J. M. and D. H. Reiley (2012, April). The economics of spam. *Journal of Economic Perspectives 26*(3), 87–110.

Schneier, B. (2008). Inside the twisted mind of the security professional. *Wired Magazine*.

Schneier, B. (2017). Inside the twisted mind of the security professional. *Edge*.

Schryen, G. (2009). A comprehensive and comparative analysis of the patching behavior of open source and closed source software vendors. In *Proceedings of the 2009 Fifth International Conference on IT Security Incident Management and IT Forensics*, IMF '09, Washington, DC, USA, pp. 153–168. IEEE Computer Society.

Schwartz, E. J., T. Avgerinos, and D. Brumley (2011). Q: Exploit hardening made easy. In *USENIX Security Symposium.*

Serra, E., S. Jajodia, A. Pugliese, A. Rullo, and V. Subrahmanian (2015). Pareto-optimal adversarial defense of enterprise systems. *ACM Transactions on Information and System Security (TISSEC) 17*(3), 11.

Sheyner, O., J. Haines, S. Jha, R. Lippmann, and J. M. Wing (2002). Automated generation and analysis of attack graphs. *Proceedings of the 29th IEEE Symposium on Security and Privacy 0*, 273.

Stock, B., S. Lekies, and M. Johns (2013). 25 million flows later-large-scale detection of dom-based xss. *20th CCS. ACM*.

Stokey, N. L. (2008). *The Economics of Inaction: Stochastic Control models with fixed costs.* Princeton University Press.

Stone-Gross, B., M. Cova, L. Cavallaro, B. Gilbert, M. Szydlowski, R. Kemmerer, C. Kruegel, and G. Vigna (2009). Your botnet is my botnet: analysis of a botnet takeover. In *Proceedings of the 16th ACM Conference on Computer and Communications Security (CCS'09).* ACM.

Telang, R. and S. Wattal (2007). An empirical analysis of the impact of software vulnerability announcements on firm stock price. *Software Engineering, IEEE Transactions on 33*(8), 544–557.

Van Eeten, M. and J. Bauer (2008). Economics of malware: Security decisions, incentives and externalities. Technical report, OECD.

Van Eeten, M., J. Bauer, H. Asghari, S. Tabatabaie, and D. Rand (2010). The role of internet service providers in botnet mitigation: An empirical analysis based on spam data. Technical report, OECD STI Working Paper.

Verizon (2011). 2011 data breach investigation report. Technical report, Verizon.

Wang, J. A., H. Wang, M. Guo, and M. Xia (2009). Security metrics for software systems. In *Proceedings of the 47th Annual Southeast Regional Conference*, ACM-SE 47, New York, NY, USA, pp. 47:1–47:6. ACM.

Wang, L., T. Islam, T. Long, A. Singhal, and S. Jajodia (2008). An attack graph-based probabilistic security metric. In *Proceedings of the 22nd IFIP WG 11.3 Working Conference on Data and Applications Security*, Volume 5094 of *Lecture Notes in Computer Science*, pp. 283–296. Springer Berlin / Heidelberg.

Wash, R. (2010). Folk models of home computer security. In *Proceedings of the Sixth Symposium on Usable Privacy and Security.*

Yeo, M. L., E. Rolland, J. R. Ulmer, and R. A. Patterson (2014). Risk mitigation decisions for IT security. *ACM Transactions on Management Information Systems (TMIS) 5*(1), 5.

Zhuge, J., T. Holz, C. Song, J. Guo, X. Han, and W. Zou (2009). *Studying Malicious Websites and the Underground Economy on the Chinese Web*, pp. 225–244. Boston, MA: Springer US.

6. Appendix: Proofs of Propositions

6.1. Proof of Proposition 1

The objective of the Proposition is to demonstrate the solution condition for the optimal set of action times $\{T_1^*, \ldots, T_n^*\}$, which is given by the recursive derivative:

$$\frac{\partial \Pi(T_i, T_{i-1})}{\partial T_i} e^{-\delta T_{i-1}} - \delta(\Pi(T_{i+1}, T_i) - C_i) e^{-\delta T_i} + \frac{\partial \Pi(T_{i+1}, T_i)}{\partial T_i} e^{-\delta T_i} = 0 \tag{16}$$

subject to $T_0 = 0$, $T_{n+1} = \infty$, $\delta, \lambda > 0$.

Proof. Proof of Proposition 1

We solve Eq. (3) in expectations and replace Eq. (4) and $c_i(t) \approx 0$ in Eq. (3) with the abbreviations $\theta_{-1} \equiv 0$, $\theta_0 \equiv \theta_V$, and $\theta_i \equiv \theta_{V \cup \{v_1 \ldots v_i\}}$. Hence, we obtain

$$\{T_1^*, \ldots, T_n^*\} \approx \underset{\{T_1, \ldots, T_n\}}{\arg\max} \sum_{i=0}^{n} -C_i e^{-\delta T_i} + \int_{T_i}^{T_{i+1}} rN \left(\theta_{i-1} e^{-\lambda t} + (\theta_i - \theta_{i-1}) e^{-\lambda (t - T_i)} \right) e^{-\delta t} dt \tag{17}$$

We can now solve the integral by replacing $t \to z + T_i$

$$\begin{aligned}
\int_0^{T_{i+1} - T_i} rN \left(\theta_{i-1} e^{-\lambda (z + T_i)} + (\theta_i - \theta_{i-1}) e^{-\lambda (z)} \right) e^{-\delta (z + T_i)} dz = \ldots \\
= rN e^{-\delta T_i} \left(\theta_i + \theta_{i-1} e^{-\lambda T_i} - \theta_{i-1} \right) \int_0^{T_{i+1} - T_i} e^{-(\delta + \lambda) z} dz \\
= \frac{rN}{\delta + \lambda} e^{-\delta T_i} \left(\theta_i - \theta_{i-1} + \theta_{i-1} e^{-\lambda T_i} \right) \left(1 - e^{-(\delta + \lambda)(T_{i+1} - T_i)} \right)
\end{aligned} \tag{18}$$

Hence we finally obtain the following result which can be rewritten as (5)

$$\{T_1^*,\ldots,T_n^*\} = \underset{\{T_1,\ldots,T_n\}}{\arg\max} \sum_{i=0}^{n} e^{-\delta T_i} \left(-C_i + \frac{rN}{\lambda+\delta} \left(\theta_i - \theta_{i-1} + \theta_{i-1} e^{-\lambda T_i}\right) \left(1 - e^{-(\lambda+\delta)(T_{i+1}-T_i)}\right) \right) \tag{19}$$

To identify the optimal T_i we take the usual first order condition and obtain for $i = 1 \ldots n$

$$\begin{aligned}
\frac{\partial \Pi}{\partial T_i} =& \frac{\partial}{\partial T_i}(\ldots + (\Pi(T_{i-1+1}, T_{i-1}) - C_{i-1})e^{-\delta T_{i-1}} + (\Pi(T_{i+1}, T_i) - C_i)e^{-\delta T_i} \\
&+ (\Pi(T_{i+1+1}, T_{i+1}) - C_{i+1})e^{-\delta T_{i+1}} + \ldots) \\
=& \ldots + \frac{\partial}{\partial T_i}\left((\Pi(T_i, T_{i-1}) - C_{i-1})e^{-\delta T_{i-1}} + (\Pi(T_{i+1}, T_i) - C_i)e^{-\delta T_i}\right) + \ldots \\
=& \frac{\partial \Pi(T_i, T_{i-1})}{\partial T_i} e^{-\delta T_{i-1}} - \delta(\Pi(T_{i+1}, T_i) - C_i)e^{-\delta T_i} + \frac{\partial \Pi(T_{i+1}, T_i)}{\partial T_i} e^{-\delta T_i}
\end{aligned} \tag{20}$$

End of proof. □

6.2. Proof of Corollary 1

Corollary 1 outlines the solution space when the attacker presumes the residual income after T_1^* is fixed in time t expectations, hence the attacker is myopic to action time T_1^*.

Proof. Proof of Corollary 1 For $n = 1$ Eq. (7) can be simplified as follows by substituting $T_0 = 0$ and $T_{n+1} = \infty$ and $T_n = T$.

$$\frac{\partial \Pi(T,0)}{\partial T} - \delta(\Pi(\infty, T) - Cv|V)e^{-\delta T} + \frac{\partial \Pi(\infty, T)}{\partial T} e^{-\delta T} = 0 \tag{21}$$

To determine the three components of the equation above we now decompose each of the individual terms of Eq. (6) as follows:

$$\Pi(T,0) = \frac{rN}{\lambda+\delta}\theta_V \left(1 - e^{-(\lambda+\delta)T}\right) \quad \text{and} \quad \Pi(\infty, T) = \frac{rN}{\lambda+\delta}\left(\theta_{V\cup\{v\}} - \theta_V + \theta_V e^{-\lambda T}\right) \tag{22}$$

we can now derive the partial derivatives

$$\frac{\partial \Pi(T,0)}{\partial T} = rN\theta_V e^{-(\lambda+\delta)T} \quad \text{and} \quad \frac{\partial \Pi(\infty,T)}{\partial T} = -rN\frac{\lambda}{\lambda+\delta}\theta_V e^{-\lambda T} \tag{23}$$

We then replace the corresponding value in the Eq. (21) above:

$$\begin{aligned}
\frac{\partial \Pi}{\partial T} &= rN\theta_V e^{-(\lambda+\delta)T} - \delta\left(\frac{rN}{\lambda+\delta}\left(\theta_{V\cup\{v\}} - \theta_V + \theta_V e^{-\lambda T}\right) - C(v|V)\right)e^{-\delta T} - rN\frac{\lambda}{\lambda+\delta}\theta_V e^{-\lambda T} e^{-\delta T} \\
&= \frac{rN}{\lambda+\delta} e^{-\delta T}\left((\lambda+\delta)\theta_V e^{-\lambda T} - \delta\left(\left(\theta_{V\cup\{v\}} - \theta_V + \theta_V e^{-\lambda T}\right) - (\lambda+\delta)\frac{C(v|V)}{rN}\right) - \lambda\theta_V e^{-\lambda T}\right)
\end{aligned} \tag{24}$$

which finally yields

$$\frac{\partial \Pi}{\partial T} = rNe^{-\delta T}\left(\frac{C(v|V)}{rN} - \frac{\delta}{\lambda+\delta}(\theta_{V\cup\{v\}} - \theta_V)\right) \tag{25}$$

Now observe that if $\frac{C(v|V)}{rN} > \frac{\delta}{\lambda+\delta}(\theta_{V\cup\{v\}} - \theta_V)$ then the derivative is positive decreasing and it is it is convenient to postpone the update which is eventually reached for $T^* \to \infty$. This is particularly true when $\theta_{V\cup\{v\}} - \theta_V = 0$ and namely there is no change in the number of infected systems by adding one more vulnerability.

If $\frac{C(v|V)}{rN} \leq \frac{\delta}{\lambda+\delta}(\theta_{V\cup\{v\}} - \theta_V)$ the derivative is negative so any update would decrease the marginal return. Only if $\frac{C(v|V)}{rN} = \frac{\delta}{\lambda+\delta}(\theta_{V\cup\{v\}} - \theta_V)$ then the derivative is identically zero and the attacker is indifferent to the time of deployment. End of proof. □

6.3. Proof of Proposition 2

The final theoretical result provides an explicit solution to the myopic version of the dynamic programming problem and proceeds as follows:

Proof. We impose the stopping condition to the first order derivative of the profit of the attacker in Eq. (25)

$$
\begin{aligned}
\frac{\partial \Pi}{\partial T} &= rNe^{-\delta T}\left(\frac{C(v|V)}{rN} - \frac{\delta}{\lambda+\delta}(\theta_{V\cup\{v\}} - \theta_V)\right) = \frac{r(0, N_V(0), V)}{N_V(0)} = r \\
& \quad N\left(\frac{C(v|V)}{rN} - \frac{\delta}{\lambda+\delta}(\theta_{V\cup\{v\}} - \theta_V)\right) = e^{\delta T} \\
T_r &= \frac{1}{\delta}\log\left(\frac{C(v|V)}{r} - \frac{\delta}{\lambda+\delta}(\theta_{V\cup\{v\}} - \theta_V)N\right) \qquad (26)
\end{aligned}
$$

As the exploit weaponization has to happen for $T_r \geq 0$ we must have $\frac{C(v|V)}{r} - \frac{\delta}{\lambda+\delta}(\theta_{V\cup\{v\}} - \theta_V)N \geq 1$ and therefore $C(v|V) \geq r + \frac{\delta}{\lambda+\delta}(\theta_{V\cup\{v\}} - \theta_V)rN$. End of proof. □

6.4. Proof of Claim 1

The transformation of the model prediction to the number of attacks against $\theta_{V\cup\{v\}}N$ systems received $\mathcal{T}$ days after receiving an attack against a different vulnerability is as follows.

Proof. Setting the number of attacks on the first vulnerability at time $t - \mathcal{T}$ as $\mathcal{U}(\theta_v, t - \mathcal{T}) = N\theta_v e^{-\lambda(t-\mathcal{T})}$ and the attacks received on the second vulnerability at time t as $\mathcal{U}(\theta_{V\cup\{v\}}, t) = N\theta_{V\cup\{v\}}e^{-\lambda(t-T^\star)}$, we obtain that the expected attacks received $\mathcal{T}$ days after the first attack are as follows:

$$
\begin{aligned}
\mathcal{U}(\theta_{V\cup\{v\}}, t, \mathcal{T}) &= \int_{\max(\mathcal{T},T^\star)}^{+\infty} \min\left(N\theta_v e^{-\lambda(t-\mathcal{T})}, N\theta_{V\cup\{v\}}e^{-\lambda(t-T^\star)}\right) dt \\
&= \min(N\theta_v e^{\lambda\mathcal{T}}, N\theta_{V\cup\{v\}}e^{\lambda T^\star}) \int_{\max(\mathcal{T},T^\star)}^{+\infty} e^{-\lambda t} dt \\
&= \frac{1}{\lambda}\min(N\theta_v e^{\lambda\mathcal{T}}, N\theta_{V\cup\{v\}}e^{\lambda T^\star})e^{-\lambda(\max(\mathcal{T},T^\star))} \\
\log\mathcal{U}(\theta_{V\cup\{v\}}, t, \mathcal{T}) &= \log\frac{1}{\lambda} + \min(\log N\theta_v + \lambda\mathcal{T}, \log N\theta_{V\cup\{v\}} + \lambda T^\star) - \lambda(\max(\mathcal{T}, T^\star))
\end{aligned}
$$

Solve for the case $T^\star > \mathcal{T}$.

As $N\theta_v \leq N\theta_{V\cup\{v\}}$, we have that $\min(\log N\theta_v + \lambda\mathcal{T}, \log N\theta_{V\cup\{v\}} + \lambda T^\star) = \log N\theta_v + \lambda\mathcal{T}$, and we obtain:

$$
\begin{cases}
T^\star > \mathcal{T} \\
\log\mathcal{U} = \log\frac{N}{\lambda} + \log\theta_v + \lambda\mathcal{T} - \lambda T^\star
\end{cases} \qquad (27)
$$

Solve for the case $T^\star \leq \mathcal{T}$.

Case 1. For $\log N\theta_v + \lambda\mathcal{T} \leq \log N\theta_{V\cup\{v\}} + \lambda T^\star$ we obtain:

$$
\begin{cases}
T^\star \leq \mathcal{T} \leq \frac{1}{\lambda}\log\frac{\theta_{V\cup\{v\}}}{\theta_v} + T^\star \\
\log\mathcal{U} = \log\frac{N\theta_v}{\lambda}
\end{cases} \qquad (28)
$$

which indicates that, within a small timeframe after the introduction of the exploit at time $T^\star$, the number of received attacks only depends on the number of vulnerable systems in the wild. This result appears to explain the observation noted in Mitra and Ransbotham (2015).

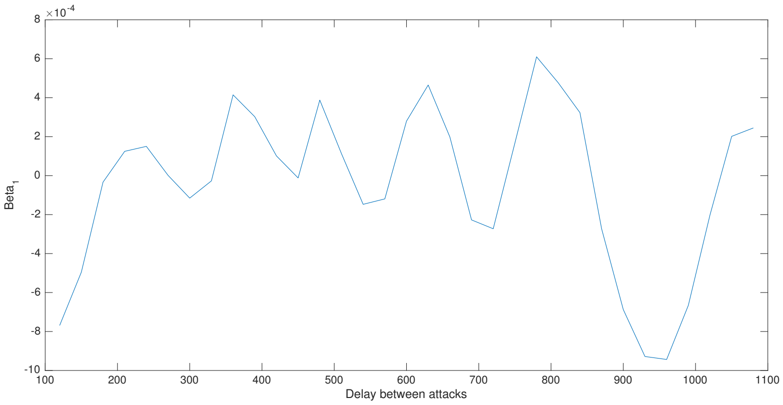

Figure 7: β_1 estimations for increasing values of $\mathcal{T}$.

Case 2. For $\log N\theta_v + \lambda\mathcal{T} > \log N\theta_{V\cup\{v\}} + \lambda T^\star$ we obtain:

$$\begin{cases} \mathcal{T} > \frac{1}{\lambda}\log\frac{\theta_{V\cup\{v\}}}{\theta_v} + T^\star \\ \log\mathcal{U} = \log\frac{N}{\lambda} + \log\theta_v + \log\theta_{V\cup\{v\}} - \log\theta_v + \lambda T^\star - \lambda\mathcal{T} \end{cases} \tag{29}$$

By comparing Eq. 27 with Eq. 29 it is apparent that the impact of $\mathcal{T}$ on $\log\mathcal{U}$ should change in sign relative to $\mathcal{T}$. Figure 7 plots the estimated coefficient β_1 for the empirical variable $\mathcal{T}$ by constraining intervals of 120 days (four months) for increasing values of $\mathcal{T}$ (e.g. $0 < \mathcal{T} \leq 120,\ 30 < \mathcal{T} \leq 150,\ 60 < \mathcal{T} \leq 180,\ \ldots$). It can be observed that the sign of β_1 oscillates above and below zero with increasing amplitude as $\mathcal{T}$ increases. This effect is present regardless of the size of the interval and the relative increment imposed on $\mathcal{T}$.

$\square$

7. On-line Appendix: Replication Guide

Here we report the replication guidelines for our study and detail the construction of the tables from WINE, our data can be reproduced by utilizing the reference data set *WINE-2012-008*, archived in the WINE infrastructure.

Basic Tables from WINE

The first table we construct from WINE, LIFE, reports the full history of attacks against distinct users in WINE. It has the following structure:

LIFE =

UserID	AttackSignature	Date	SystemID	IP_HASH	VolumeAttacks
⋮	⋮	⋮	⋮	⋮	⋮

where UserID is the unique identifier of a user in the WINE platform, and AttackSignature is the unique ID identifying the signature that the attempted attack triggered. Date, SystemID, IP_HASH report respectively the day, month and year of the attack; the internal ID of the operating system build; the hash of the IP address. Finally, VolumeAttacks reports how many attacks UserID received on that day.

The attack profile defined in the LIFE table may depend on the interaction between several factors.

In particular, we identify three main factors that may confound our observations: the platform on which the attacked user operates; his/her geographical location; the user evolution. To control for these factors, we extract two additional tables:

PLATFORM =

UserID	SystemID	OperatingSystem	Version	ServicePack
⋮	⋮	⋮	⋮	⋮

PLATFORM links a UserID to the type of system installed on the machine. All systems considered in this study are running on Microsoft Windows.

TARGET_PROFILE =

UserID	VolumeAttacks	Day	Month	Year
⋮	⋮	⋮	⋮	⋮

In the TARGET_PROFILE table we record the volume of attacks that a UserID receives in a day, irrespectively than the used platform or IP address.

STABILITY =

UserID	UserID	Country	SystemID
⋮	⋮	⋮	⋮

In the STABILITY table we record the different countries from which a specific SystemID connected. This can also be obtained by aggregating the data in the table LIFE.

USERS =

UserID	User Profile	Frequency	Peak
⋮	⋮	⋮	⋮

Finally, we aggregate the data in these two tables in a third extracted table USERS, in which we categorise each user in WINE over three dimensions: User Profile, Frequency, and Peak. The first dimension records whether the user changes country and/or updates his or her system within the lifetime of WINE. In Frequency and Peak we record respectively the frequency of received attacks and the maximum volume of attacks received in a day by the user. Aggregated forms of USERS, PLATFORMS, TARGET_PROFILE and STABILITY are disclosed.

Data merging and aggregation

Our final dataset is obtained by first joining the LIFE table with the control tables PLATFORM, USERS, TARGET_PROFILE, and then by performing a self-join of the obtained table with itself. The goal of the self-join is to obtain the pairs of *subsequent* attack signatures triggered by a single user, and the time passed in between the two attacks. The final disclosed table is of the form:

SIGNATURES =

SID_1	SID_2	$\mathcal{T}$	$\mathcal{U}$	$\mathcal{N}$	Country	OS	OS_V	OS_SP	User Profile	Frequency	Peak
⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮	⋮

where SID_1=SignatureID and SID_2=SignatureID_later identify respectively the attack signature triggered in the first and the second attack received by the user. $\mathcal{T}$ reports the days passed in between the two attacks. $\mathcal{U}$=count(machineID) reports the number of machines affected by SignatureID_later $\mathcal{T}$ days after receiving an attack of type SignatureID. Similarly, $\mathcal{N}$=sum(volumes of SignatureID_later) counts how many times these two attacks triggered an alarm $\mathcal{T}$ days apart. The difference with the previous field is that single machines may receive more than one attack of this type, thus we have sum(volumes of SignatureID_later) $\geq$count(machineID). The remaining fields are obtained with a join with the tables PLATFORM, USERS, TARGET_PROFILE.

Merging WINE with CVEs

SignatureID and SignatureID_later are internal codes that identify which attack signature deployed by Symantec's product the attack triggered. To identify which vulnerability (if any) the attack attempted to exploit we map WINE's SignatureID with the threat description publicly available at Symantec's Security Response dataset.[12] In the attack description it is provided, when relevant, the vulnerability that the attack exploits, as referenced by the unique CVE vulnerability identifier.[13] The CVE-ID is a standard reference identifier for software vulnerabilities introduced by the MITRE organization and used by all major vulnerability databases such as the National Vulnerability Database, NVD.[14]

To characterize each vulnerability, we match the CVE-ID reported in the SignatureID with the vulnerability summary reported in the NVD. The information on NVD comprises the name of the affected software Sw (e.g. Flash in the example above), the latest vulnerable version Ver of the software (in our example Flash 9.0.115 and 9.0.45), and the disclosure date Day.

Further, additional information describing the technical aspects of the vulnerability are also provided. This information can be extracted from the Common Vulnerability Scoring System (CVSS) assessment of the vulnerability. CVSS measures several technical dimensions of a vulnerability to obtain a standardized assessment that can be used to meaningfully compare software vulnerabilities. However, previous studies showed that not all measures show, in practice, enough variability to characterize the vulnerabilities Allodi and Massacci (2014). Of the dimensions considered in CVSS, in this study we are specifically interested in the Access Complexity and Imp measures. The former gives an assessment on the 'difficulty' associated with engineering a reliable exploit for the vulnerability Mell et al. (2007). For example, a vulnerability that requires the attacker to win a race condition

[12] The reference dataset can be found at https://www.symantec.com/security_response/, last accessed January 15, 2017.

[13] The classifiers are available at http://cve.mitre.org, last accessed January 15, 2017.

[14] Full database can be found at http://nvd.nist.gov, last accessed January 15, 2017.

on the affected system in order to successfully exploit kit may be deemed as a High complexity vulnerability (because the attacker can not directly control the race condition, thus exploitation can be only stochastically successful). Similarly, `Imp` gives an assessment on the Confidentiality, Integrity and Availability losses that may follow the exploitation of the vulnerability.