Competitive and strategic effects in the timing of patch release

Ashish Arora, Christopher M. Forman, Anand Nandkumar[1] and Rahul Telang
{ashish,cforman, anandn,rtelang}@andrew.cmu.edu
Carnegie Mellon University

First Draft: October 2005
This version: May 2006

Abstract:

The relationship between quality and market concentration has long been of interest to both policy makers and economists. In our application, we focus on the effect of competition on one aspect of software quality – time taken by software vendors to release patches that fix vulnerabilities. We empirically estimate how the extent of competition affects the timing of patch release using a novel dataset assembled for the purposes of this research. Competition in the context of information security has two separate effects: First is the disclosure threat effect - the possibility of any of the other vendors that are also affected by the same vulnerability, releasing patch earlier, thereby implicitly disclosing the vulnerability. Second is the competition effect, which is the effect of end users penalizing laggards, by comparing responses of vendors that also sell a similar product. Our results suggest disclosure threat hastens the arrival of a patch by about 24 days, whereas competition effect hastens patch release by 52 days on average. Further, firms with larger sales (of the product) patch faster: a 10% increase in firm sales is associated with an earlier patch release by about 2.5 days. Therefore, our results support the notion that greater competition results in better ex-post service quality. Further, to the extent that earlier patch release minimizes consumer loss, our results suggest that when higher number vendors are affected by the same vulnerability end user losses are lower, thereby leading to better social outcomes.

Keywords: Vulnerability disclosure, quality, competition.

[1] Corresponding author

1. Introduction:

According to CERT/CC, the number of software vulnerabilities[2] reported to CERT/CC in year 2005 alone was about 5990. The rapid increase in the number of vulnerabilities discovered in software over the past few years has strengthened the argument that poor software quality of software is an artifact of high concentration that is typical of most software markets. Of particular concern is the potential for firms with market power to deliberately under provide quality, in our context ex post support. This can occur due to a variety of reasons: First, information goods industries like software, offer significant early mover advantages, resulting in incentives to release products earlier despite the product not being "ready" for release. Although, early release would also require substantial investments in ex post product support as pointed out by Arora, Caulkins and Telang (2005) investment in ex post support could also depend on the firms' market power. Arguably higher amounts of concentration could result in lower investments in ex-post support. Specifically, firms with market power could deliberately under provide quality in an effort to maximize profits again implying under-investment in ex-post support. Second, lack of user willingness to pay for software quality could also result in lower investment in ex-post product support. If poor product quality is due to high market concentration then introducing more competition should lead to higher quality of software, *ceteris paribus*. Conversely if under provisioning of quality is due to lack of user willingness to pay then competition should have little impact on quality. In this paper we provide empirical evidence on the impact of competition on vendors' incentives to release the patches faster.

The relationship between quality and market concentration has long been of interest to both policy makers and economists. Our focus is on the effect of competition on one aspect of software quality – time taken by software vendors to release patches that fix vulnerabilities. We empirically estimate how the extent of competition affects the timing of patch release using a novel dataset comprising of software vulnerabilities reported to CERT/CC and vendors response to such vulnerabilities. The time taken by vendors to release patches upon discovery of a vulnerability as a measure of software quality (security) because (i) The timely availability of patches critically determines the amount of losses incurred by consumers and hence the value consumers derive from the software product. Therefore, the timing of patch is very similar in nature to ex-post service quality (ii) Since software intrinsically require patches for quality

[2] A vulnerability is a software bug that can be taken advantage of by an attacker to compromise an end user's computer.

upgrade or to fix vulnerabilities (temporary quality degradation) timing of patch release is often viewed by end users as an important element of ex post product support (iii) Unlike many other measures of quality, timing of patch release is more reliably measured.

Given the rapid increase in the number of reported software vulnerabilities[3] and the consequent economic damages to end users, the factors that contribute to the timing of vendors' patch release has been a matter of great interest among members of the software community. End users suffer losses from software vulnerabilities when malicious users or "attackers" take advantage of vulnerabilities to inflict monetary losses[4] to end users. Patches released by vendors enable end users to prevent attackers from exploiting these vulnerabilities. The sooner the patch is released, the lower are customer losses. Many members of the security community have recommended regulation aimed at providing incentives for software vendors to minimize the time window of exposure to end users. However the type of regulation that would minimize social losses from vulnerabilities would critically depend upon proper understanding of factors that condition the timing of patch release to vulnerabilities. Despite its importance this is an area that has lacked empirical evidence mainly due to the non-trivial effort involved in data collection. Although the focus of this work is to understand how the timing of patches varies with the number of competitors, we also provide valuable empirical evidence on the factors that condition the timing of vendors' patch release.

Our results suggest that competition results in better quality as evidenced by quicker time to release patches by vendors to the extent of about 52 days on an average. Interestingly, our results also brings to focus another important issue of "threat of disclosure" (disclosure threat hereafter) which is the issue of how public disclosure of a flaw that is common to many products affects vendors' responses to vulnerabilities. Our results suggest that disclosure threat results in an earlier patch release by about 24 days. These results demonstrate that competition does have an effect on ex post product support provided by software vendors.

The rest of the paper is structured as follows: We first provide some background on the domain of software vulnerabilities and highlight key problems in estimating the impact of competition on the timing of patch release in section 2. In the process we also provide intuition for our

[3] According to CERT/CC, the number of vulnerabilities reported in year 2005 alone was about 5990.
[4] These monetary losses are typically due to loss of confidential information that might be stolen by attackers, losses due non-availability of computing infrastructure as a result malicious use by attackers and loss of integrity of information as a result of attack.

identification strategy. Section 3 reviews related literature. Section 4 provides a description of the data sources used in empirical analysis. We conclude and discuss limitations of our results in section 7.

2. Software vulnerabilities and patches

Unlike many physical goods, problems related to software can be mitigated even after product release. This makes patches an intrinsic part of any software life cycle. Vendors try to introduce the product relatively early in the product development cycle even though early product release might entail greater investments in ex post support (Arora, Caulkins and Telang 2005). This makes both vulnerabilities in software as well as patches that fix vulnerabilities intrinsic to any "shrink wrapped" software. Patches are also perceived by end users as a very important component of ex-post product support and an important signal of quality. This is because the probability of a malicious attacker exploiting a specific vulnerability to compromise end user computers is positively correlated with the amount of time the vulnerability remains without a fix. Thus, the timing of patches critically determines the extent of end user losses. In the absence of other vendors also being affected by the same vulnerability, two considerations drive the timing of the vendor's patch – (i) the extent to which end user losses affect the future demand for the product and, (ii) the cost to fix the vulnerability. Typically, an early fix entails higher costs but also reduce customer losses and, hence also, reduce reputation loss and loss of future sales. Vendors, thus choose an optimal time to release patch to minimize their total losses.

In many cases, a newly discovered vulnerability could affect many different products ("common" vulnerability hereafter). A common vulnerability is typically a result of a shared code base or design specification, or, due to a proprietary extension of a widely used software component. When vendors share a common vulnerability, vendors are still driven by the considerations outlined above to determine the optimal time to release patch although there are some important additional considerations. When a vulnerability is known to be common to many products, if one vendor releases patch for its product, it implicitly publicly discloses the vulnerability in the products manufactured by other vendors (that also share the same vulnerability). This presumably results in higher end user losses to other vendors. In short, presence of many vendors for a vulnerability acts as a disclosure threat. Presumably, higher disclosure threat would shrink the time to patch (we call this as disclosure effect). Also, the literature on product quality and competition (reviewed below) suggests that when there are many competing products, end users have more choices, and thus, future sales of a product are likely to be more sensitive to perceived quality. In our context, this implies that end users can compare vendor responses and penalize laggards (competition effect). This view is consistent with arguments put forth by researchers understanding similar issues in other industries, e.g. Suzuki (2000); and Cohen and Mazzeo (2004). Thus presence of many vendors may also reduce the patching time (because of "competition effect"). Note that competition effect emanates only from vendors that also operate in the same product market or sell a similar product (we call such vendors as rivals). However disclosure threat effect emanates from all vendors that are affected by the common vulnerability regardless of whether they are rivals. In the paper we identify these effects separately and show how competition and disclosure threat effects influences vendors' time to patch vulnerabilities.

3. Literature review:

Our work draws from two varied streams of literature – literature on competition and quality and literature on information security specifically related to vulnerability disclosure.

A. Competition and quality

From a theoretical perspective the relationship between quality and competition has been well studied. Swan (1970) argued that quality (interpreted in terms of durability of a product) was independent of market structure. This independence result intrigued many researchers and many subsequent works reversed the independence finding. Spence (1975) for instance argued that an unregulated monopolist's provisioning of quality is likely to be biased away from the social optimum. A monopolist would provide optimum quality only when elasticity does not vary with quality. Levhari and Peles (1973) show that when quality is a substitute for quantity, both quality and quantity provided by the monopolist might fall short of those in the competitive market. Gal-Or (1983) shows that the average quality in a market actually declines as a result of additional entry. This is because upon entry, the ability of a single firm to segment the market declines. Hence firms on an average produce more quantity of low quality goods to discriminate more effectively among consumers that highly value the product. Schmalensee (1979) notes that the outcomes of the different theoretical models are very sensitive to the assumptions made by the models and articulates the need for empirical evidence to probe the implications theoretical work in this area.

Understandably, given that measuring quality in an unambiguous manner is non-trivial, empirical work on the relationship of quality and competition is not as widespread. Demberger and Sherr (1989) provide evidence that deregulation in the legal services industry leads to greater customer satisfaction. Dranove and White's (1994) studied the issue of quality and competition in hospital markets and suggested that higher market concentration invariably leads to lower quality in hospital markets. Borenstein and Netz (1999) note that airlines were less likely to schedule their flights at passengers' most preferred times during the period of price regulation. Hoxby (2000) found that metropolitan areas with more schools districts produce higher quality measured in terms of student achievements. Mazzeo (2003) provides evidence of longer flight delays in more concentrated airline markets. Cohen and Mazzeo (2004) in an analysis of the banking industry find evidence of higher quality (measured in terms of number of branches) when banks face multi-market banks as competitors as opposed to when banks face single-market banks as competitors.

In our application, we use time to release patch of a vulnerability upon its discovery as our measure of quality to examine the quality-competition relationship thereby providing evidence about this relationship in the context of software industry. As with other empirical studies in this area, our conclusions also support the notion that higher competition is associated with an earlier patch release and hence better quality.

B. Economics of information Security

Researchers that work in area of economics of information security especially recently have concerned themselves in understanding how vulnerability information disclosure affects social loss from software vulnerabilities. Schneier (2000) argued that the loss from attacks, are not only influenced by the intensity of attacks, but also on how long the vulnerability remains un-patched. But attack intensity may also depend on whether the vulnerability is public information as pointed out by Arora, Nandkumar and Telang (2004). In this context, Arora, Telang and Xu (2004), develop a theoretical model to examine how an optimal disclosure policy can influence the behavior of vendors and reduce the social cost of vulnerabilities. In particular, they show that early disclosure of vulnerabilities is not necessarily socially optimal, though it will result in the vendor releasing patch earlier. They consider the roles of three players, the vendor, the customers and the social planner. The vendor trades off between the cost to develop patch and customer loss. The customer suffers breaches as a result of being exposed to the vulnerability, especially when no patch exists for the vulnerability. Since disclosure policy affects the vendor and customers in conflicting ways, the role of a social planner becomes one of designing an optimal policy such that the vendors provide a patch fast enough, at a reasonable cost. Cavusaglu et al (2005) model the multiple vendor case and show how a policy maker should set policies. Nizovtsev and Thursby (2005) examine the factors that influence a benign identifiers' decision to disclose vulnerabilities. They show that the current situation with regard to disclosure constitutes mixed strategy equilibrium of a game in which the benign identifiers play the role of loss minimizing agents. Choi, Fershtman and Gandal (2005) examine how vulnerabilities affect vendors and how consumers buy software. They conclude that vendors are likely to announce vulnerabilities when the probability of an attacker exploiting a vulnerability is relatively high and that it is possible for vendors to announce vulnerabilities even if it is not socially optimal to announce them.

The empirical stream of literature relating to vulnerability disclosure however is relatively sparse. Arora, Nandkumar and Telang (2004) provide empirical evidence on the impact of publication of vulnerabilities when disclosure is not accompanied by patches. They find that undisclosed vulnerabilities attract the least number of attempts to breach a host, while vulnerabilities that are disclosed without a patch attract the most number of attempts to breach a host. To the extent that such breaches are correlated with monetary losses early disclosure could result in substantial economic losses[5]. Arora, Krishnan, Telang and Yang [2005] using a dataset assembled from CERT/CC's vulnerability notes and SecurityFocus database, conclude that early disclosure influences the vendor to release patch earlier with vulnerability disclosed by CERT/CC being patched faster by vendors. Telang and Wattal (2004), find empirical evidence of firms' incurring loss in market value, as a result of vulnerability disclosure. To our knowledge, the issue of vendors' response to common software vulnerabilities has not been studied by researchers and hence marks a contribution to this literature.

4. Data and variables:

The sample for the purposes of this study was constructed from two sources. We acquired details of software vulnerabilities from CERT/CC. Variables related to number of customer (or quantity) was acquired from the Harte Hanks database.

4.1. Vulnerability data:

[5] The CSI-FBI survey 2004, estimated the annual losses from information security incidents to be about $1.4 billion in 2004 alone

Variables that relate to vulnerabilities were assembled from vulnerability publications of CERT/CC. This data source lists all the vulnerabilities that were reported to it, the vendor-product combination that was affected by the vulnerability, along with the date of publication of the vulnerability. A typical vulnerability reporting process is as follows: An identifier reports the presence of a vulnerability to CERT/CC. CERT/CC researches the vulnerability before contacting the vendor and does so only if the presence of the vulnerability is authentic. Once the vendor is notified of an existence of the vulnerability, the vendor may choose either to respond or not to respond to CERT/CC's information of the vulnerability. Should the vendor decide to respond to CERT/CC's information, the a typical response takes one of the following forms: (i) vendor acknowledges of the presence of the vulnerability, in which case, the CERT/CC provides a specific time window to release a patch for the vulnerability (ii) vendor responds by contending that product(s) in question is not vulnerable, in which case CERT/CC just lists the vendor as not being vulnerable or as vulnerable without a patch. These responses are captured in the dataset as "status vulnerable" and "status not vulnerable" respectively. In a case where the vendor chooses not to respond to the vulnerability, then CERT/CC records the vendor's response as "status unknown". On an average, in a year, about 3000 vulnerabilities get reported to CERT/CC of which only about 10% are published.

Our unit of observation is the vendor – vulnerability pair. From September 2000 to August 2003, CERT/CC published a total of 526 vulnerability notes. A total of 622 different vendors were affected by these vulnerabilities. In all, these constituted about 4659 observations. Of these, 762 observations had status "not vulnerable", 2182 were status "unknown" while 1714, had status "vulnerable". We retained only observations with "status vulnerable" for the purpose of empirical analysis.

From these, we dropped observations wherein the vendors discovered and disclosed the vulnerability to CERT/CC of its own accord along with a patch. We also dropped observations that represented open source vendors (since these vendors are frequently small and may not conform to standard profit maximization notions) and vendors that are not head-quartered in USA (since the competitive environment could be very different, and we are likely to have poor measures of their market share). We also removed protocol vulnerabilities from the data, as these vulnerabilities as typical fixes to such vulnerabilities involve substantial design change at the protocol level and not just at the level of a product. This provided us with a sample consisting of 241 distinct vulnerabilities and 473 observations.

4.2. Market Data:

One of our key independent variables, quantity was collated using information in Harte-Hanks database (HH database), an in-depth technology end-user database that collects detailed data software consumption pattern. From 2000-2002, the survey had responses from about 58,094 organizations in the United States. Even so, it is extremely difficult to determine the number of copies of a software product in use. Instead we use a proxy. We use the number of establishments that bought at least one copy of the product weighted by number of employees in the organization as our proxy for quantity (**QUANTITY**). For instance if 1000 establishments own at least 1 licensed copy of Red Hat Linux, and each establishment has 500 employees, our measure for quantity would be 500,000, which is the aggregate number of employees in those establishments.

Weighting the number of establishments with the number of employees in the establishment puts more weight on products used in larger organizations, and arguably provides with us a more accurate proxy for quantity. Since the HH –database over samples certain industry sectors we compared the number of establishments in the HH dataset with the number of establishments in the Census and re-weighted the number of establishments in the sample to obtain a representative sample of establishments (Please refer appendix II for a detailed description on how we re-weighted our quantity measure). This procedure was adapted from the procedure followed by Forman, GoldFarb and Greenstein (2005).

Table 1 Description of variables

Variable	Description
DURATION	Time taken by vendors to patch vulnerabilities
LOGDURATION	Log of DURATION
VENDORS	Total number of vulnerable vendors
LOGVENDORS	Log of 1+**VENDOR**
INSTANT	Instant disclosure
NONINSTANT	Non-instant disclosure
SUPPLIER	=1 if vendor is a supplier of a software component that is used by another downstream vendor.
QUANTITY	Total # of employees at customers (those that used the software) sites
LOGQUANTITY	Log(1+QUANTITY)
LOGVERSIONS	Log of number of versions
SEVERITY	CERT/CC severity metric
LOGSEVERITY	Log of CERT/CC metric.
LEADER	Vendor(s) that patches before all other vulnerable vendors.

Table 1 summarizes the key variables used in the empirical analysis. These are further discussed below.

4.3. Duration:

Our key dependent variable is **DURATION**, which measures the elapsed time in calendar days from the date when the vendor came to now of the presence of the vulnerability and when the vendor released a fix for the vulnerability. The value that **DURATION** takes depends on the regime of disclosure – *instant or non-instant disclosure*.

Scenario 1: Instantly disclosed vulnerabilities

If the vulnerability is *instantly disclosed*[6], **DURATION** is the elapsed time in days between when the vulnerability was known public and the time the vulnerability was fixed by the relevant vendor.

Scenario 2: Non-instantly disclosed vulnerabilities

If the vulnerability was *non-instantly disclosed*[7], **DURATION** is the elapsed time between when CERT/CC informed the vendor of the existence of the vulnerability and when the relevant vendor issued a patch.

Our final sample comprised of 473 observations, relating to 241 distinct vulnerabilities. Of these, about 4.2%, or about 20 observations, had no patch. [8] In our empirical analysis we use the log of the elapsed days, **LOGDURATION** as our dependent variable. For the observations for which, the vendor has not yet released a patch, we assign the highest value of the dependent variable. Our results are unchanged when we redid the analysis by using a censored regression specification by treating these observations as right censored (these results are not reported in this manuscript but can however be provided upon request).

4.4. Competition:

We use **LOGVENDORS,** which, is the natural log of the total number of vendors listed as "vulnerable" by CERT for a specific vulnerability to measure the effect of competition within a

[6] a scenario in which, CERT/CC informs the vendor of the presence of a vulnerability when the vulnerability was already known public

[7] A scenario in which, CERT/CC disclosed the presence of a vulnerability to the vendor while the vulnerability is unknown to the general public.

[8] One aspect that must be noted about the empirical setup, is that we assign a value of 8.27 (maximum value of *log(***DURATION** $_{vi}$ *)*in the sample) to the dependent variable in cases where the vulnerability was not patched by the vendor. As noted earlier, there are about 20 observations that are not patched in the entire sample.

vulnerability. Note, that **LOGVENDORS** includes all vendors that are affected by the same vulnerability. Also, this measure in itself is not sufficient to identify the effect of competition separately from the effect of disclosure threat. The method by which identify the effect of competition separately from disclosure threat is discussed later in the manuscript.

4.5. Vulnerability Severity measure:

In order to account for differences in severity of vulnerabilities we use the natural log of CERT/CC's severity measure (**LOGSEVERITY**), which is a number between 0 and 180, to capture the differences in severity of vulnerabilities.[9].

4.6. Vulnerability identifier:

In our dataset we also capture the party that identified the vulnerability first. This data was gathered from CERT's publications and other public security forums like SecurityFocus's bugtraq mailing list. We classified the identifier of the vulnerability into whether the vulnerability was first discovered by a security consulting firm (**CONSULTANT**). In the sample, about 29% of all vulnerabilities comprising of 71 distinct vulnerabilities were identified by security consulting companies. Descriptive statistics for the sample are provided in table 2, provided at the end of the paper.

<Table 2 about here>

5. Empirical evidence:

We adopt two methods to test our propositions – a comparison of sample means and a regression framework. Though any regression framework makes it convenient to add controls, it imposes functional form restrictions.

5.1. Identification of competition and disclosure threat effect using instant disclosure:

Our goal is to identify the effects of disclosure threat and competition separately, which in general is difficult. We use the disclosure regime, namely *instant* and *non-instant,* as a point of

[9] The set of criteria that determines the measure is available in CERT/CC's website. The important determinants of the measure include (i) Is information about the vulnerability widely available or known? (ii) Is the vulnerability being exploited in the incidents reported to US-CERT? (iii) Is the Internet Infrastructure at risk because of this vulnerability? (iv) How many systems on the Internet are at risk from this vulnerability? (v) What is the impact of exploiting the vulnerability? (vi) How easy is it to exploit the vulnerability? (vii) What are the preconditions required to exploit the vulnerability? See www.kb.cert.org/vuls/html/fieldhelp

leverage to identify the effects separately. Since by definition there is no threat of disclosure under *instant disclosure* the number of vulnerable vendors under *non-instant disclosure* (**LOGVENDORS*NONINSTANT**) measures the combined effects of competition and disclosure threats. The number of vulnerable rivals under *non-instant* disclosure (**LOGVENDORS*INSTANT**) provides an estimate of the effect of competition. The effect of disclosure then can be estimated by differencing the competition effect and the combined effects in a linear estimation framework such as an OLS regression. Stated otherwise, if **X** is a vector of controls, E(LOGDURATION | LOGVENDORS*NONINSTANT=1, **X**) provides an estimate of the combined effects of competition and disclosure threats. E(LOGDURATION | LOGVENDORS*INSTANT=1, **X)** provides an estimate of the effect of competition. The effect of disclosure then is E(LOGDURATION | LOGVENDORS*NONINSTANT=1, **X**) - E(LOGDURATION | INSTANT=1, LOGVENDORS, **X).**

5.2. Comparison of sample means:

We start out our empirical analysis by comparing difference in conditional means and later on in the section estimate a more formal regression model after adding other controls. Table 2 shows the results of comparison of sample means conditional on vulnerable rivals under *instant disclosure* and vulnerable vendors under *non-instant disclosure* respectively. In the analysis we assign a value of 8.27 (log equivalent of the maximum duration in the sample) to the 20 observations for which a patch was not released by the vendor. We categorize **VENDORS** as **HIGH** if the number of VENDORS for a vulnerability was above the median and **LOW** otherwise. The difference in sample means of **LOGDURATION** between categories, under instant disclosure identifies the competition effect (3.42 – 4.00). The differences in **LOGDURATION** between the disclosure regimes provides an estimate of disclosure threat effect, which naturally differs depending on whether the number of vendors is high or low. The point estimates (3.42-2.49 & 4.00-3.40) suggest that both competition and, disclosure threat are associated with shorter time to release patches. Specifically, increase in the number competing **VENDORS** from below the median to above the median under *instant disclosure* is associated with an earlier patch release by vendors. This suggests that higher levels of competition are associated with an earlier patch both in the **HIGH** and **LOW** categories, an increase in disclosure threats are associated with earlier patches with the disclosure threat effect being higher in the HIGH **VENDOR** category.

Table 2 Comparison of conditional means of LOGDURATION for by number of vulnerable vendors (standard errors in parentheses) (****** p < 0.01** p < 0.05*p<0.10)***

VENDOR S	**Instant disclosure (1)**	**Non instant disclosure (2)**	**Disclosure effect *(3)***
High (Above Median) (A)	3.42***(0.21) (N=114)	2.49*** (0.16) (N=106)	**-0.93***** *(0.29)*
Low (Below Median) (B)	4.00 ***(0.17) (N=177)	3.40***(0.24) (N=76)	**-0.60**** *(0.29)*
Competition effect (C)	**-0.58***** *(0.27)*	-	
Competition effect and disclosure effects (Combined effect) (D)	-	-0.91*** (0.09)	
Sample median of vulnerable vendors affected	6		

We use a similar procedure to understand the effect of quantity in table 3, and categorize **LOGQUANTITY** into **HIGH** or **LOW** categories depending on whether **LOGQUANTITY** is greater than the sample median of that variable**.** Overall, we find that increase in quantity is not associated with an earlier patch release by vendors. Since it is conceivable that higher values of **LOGQUANTITY** is correlated with higher number of versions supported by the vendor, we further explore this result by further classifying **LOGQUANTITY** into two sub-categories depending on whether the number of versions supported is one or more than one (median number of versions supported by vendors in the sample). An exploration of the sample correlations of versions with **LOGQUANTITY** further strengthens the hypothesis that higher quantity is highly correlated with higher number of versions (correlation coefficient = 0.47 when **LOGQUANTITY** = **HIGH**) while the same is not true when vendors face lower quantity (correlation coefficient = 0.06 when **LOGQUANTITY** = **LOW**).

Table 3 Comparison of conditional means of LOGDURATION by establishment and versions (standard errors in parentheses)

Quantity categories **(1)**	**Overall (2)**	**# versions=1 Versions=LOW (3)**	**# versions >1 Versions=HIGH (4)**	**Effect of versions (5)**
LOGQUANTITY =”High”	3.44 (0.13) (N=266)	3.25 (0.15) (N=189)	3.89(0.25) (N=77)	**0.64** *(0.29)*
LOGQUANTITY =”Low”	3.41 (0.15) (N=207)	3.49 (0.16) (N=179)	2.97 (0.41) (N=28)	**0.52** *(0.44)*
Average effect of quantity	**0.03** *(0.20)*	**-0.24** *(0.22)*	**0.92** *(0.48)*	
Overall impact of versions	-	**3.36 (0.11)** *(N=368)*	**3.64 (0.21)** *(N=105)*	**0.28** *(0.24)*
Median LOGQUANTITY	14.83			
Mean/Median Versions	1.63/1			

Table 3 shows that when vendors release patch for only one version (median # **VERSIONS** = 1 in the sample), higher values of **LOGQUANTITY** is associated with an earlier patch, whereas when vendors support multiple versions, higher values of **LOGQUANTITY** is associated with a later patch. Also note that in general, **HIGH** number of versions is associated with a later patch release on an average. This is consistent with anecdotal evidence that quality testing of patches on multiple software configurations consumes the most time in a patch release process. From the perspective of our empirical framework, this highlights the need to control for number of versions supported while trying to understand the effect of quantity. Since with more versions vendors would have to devote more towards testing the patch, higher number of versions supported by vendors is likely to delay patch release by vendors. Thus, in our regression specifications we use the log of the number of versions supported by vendors, **LOGVERSIONS** as one of our controls.

Table 4 **Correlation between versions and LOGQUANTITY**

	Correlation coefficient	***Mean versions***
overall	0.29 (N=473)	1.56
LOGQUANTITY=”High”	0.47 (N=266)	2.10
LOGQUANTITY=”Low”	0.06 (N=207)	1.13

5.3. Regression results:

We now turn to an OLS specification and regress **LOGDURATION** on our variables of interest (**LOGVENDORS**, **LOGRIVAL**, **LOGQUANTITY** and **LOGVERSIONS)** without any additional controls. We interact **INSTANT** with **LOGVENDOR** to understand the effect of competition. Likewise we also interact **NONINSTANT (**which is **1-INSTANT)** with **LOGVENDORS** to estimate the combined effect of competition and disclosure threat under *non-instant disclosure*. As stated earlier we instrument for **INSTANT*LOGVENDORS** using **INSTANT*LOGRIVALS**. The results of this specification are shown in table 5.

Table 5- OLS, dependent variable LOGDURATION

Dependent variable *LOGDURATION*	***Coefficient (Std. Error*[+]*)***
NONINSTANT (non-instant disclosure)	-0.42 *(0.40)*
LOGVENDORS*INSTANT	-0.17 *(0.15)*
LOGVENDORS *NONINSTANT	-0.41 *** *(0.15)*
LOGRIVALS *NONINSTANT	-
LOGQUANTITY	-0.11 ** *(0.05)*
LOGVERSIONS	0.50 *** *(0.19)*
Constant	5.49 *** *(0.76)*
N	473
# vulnerabilities	241
R-squared	0.09

**** p < 0.01** p < 0.05*p<0.10. [+]Cluster corrected on vulnerability.*

The results in column 1 suggest that the effect of competition is associated with 2% earlier patch release for a 10% increase in competitors affected, and the combined effect of disclosure threat and competition is about a 4.1% decrease in **DURATION** for a 10% increase in **VENDORS.** The separate effect of disclosure threat is about 2.4% (interpreted as the difference between vendors affected under non-instant disclosure and the number of competitors affected under instant disclosure- -4.1% + 1.7%). Moreover, a 10% higher quantity is associated with a 1.1% earlier patch release.

Since our unit of observation is a vendor vulnerability pair, it is conceivable that observations differ due to reasons associated with either vulnerabilities or vendors. To control for observable differences between vulnerabilities we use two sets of measures - One, a severity identifier, **LOGSEVERITY,** to control for relative severity between vulnerabilities. Two, we also use market fixed effects to control for differences in complexity between software categories. The market fixed effects also controls for the fact that certain markets inherently may be more sensitive to vulnerabilities. It is plausible that vendors in such markets internalize a greater proportion of end user losses. Since about 94% of the sample comprised of operating system and web browser vulnerabilities we also use two market dummies[10]. Further, we also account for differences between vendors using the following controls: Vendor fixed effects to control for unobserved vendor-related factors (specifically, vendor fixed effects consist of dummy variables Apple, Compaq, SGI, HP IBM, Mandrake, Microsoft, Red Hat, SUSE, Sun, Oracle SCO) and **SUPPLIER**, to control for vendors that are primarily component providers (**SUPPLIER** takes a value of 1 if the vendor is a supplier of a software component that in turn is used in another product. For example, Macromedia Inc. that supplies a flash player plug-in component for Netscape navigator and internet Explorer is an example of a supplier). We also control for unobserved differences between vulnerabilities (like possible monetary damages that can accrue should malicious attackers succeed in compromising host using a vulnerability, or, the complexity in fixing the vulnerability) using a random effects specification. As in the sample means analysis, we assign a value of 8.27 (log equivalent of the maximum duration in the sample) to the dependent variable in cases where the vulnerability was not patched by the vendor.

[10] Vulnerabilities that were neither operating system nor web browser vulnerabilities include antivirus, Application Development, Application Server Software, Backup and Recovery software, database Management, Email software, Groupware, Lan OS, Suites, System Utilities, System software, Web Design Tools and Web Server software.

Table 6- Random effects model, dependent variable LOGDURATION

Dependent variable **LOGDURATION**	**Coefficient (Std. Error)**	
NONINSTANT	-0.47 *(0.39)*	
LOGVENDORS*INSTANT	-0.31 *(0.17)*	*
NONINSTANT*LOGVENDORS	-0.45 *(0.01)*	***
INSTANT*LOGRIVALS	-	
LOGSEVERITY	-0.16 *(0.13)*	
LOGQUANTITY	-0.15 *(0.05)*	***
LVERSIONS	0.44 *(0.20)*	**
SUPPLIER	-0.94 *(0.91)*	
Constant	7.12 *(0.90)*	***
Vendor fixed effects(11)	Yes	
Market fixed effects(2)	Yes	
R^2 (overall)	0.18	
σ_u	1.73	
N	473	
# vulnerabilities	241	

**** $p < 0.01$** $p < 0.05$*$p<0.10$*

We use the results shown in column 1 of table 7 to understand the effects of interest. A 10% increase in competitors affected is associated with a 3.1% decrease in the mean time to patch vulnerabilities while a 10% increase in disclosure threat (calculated as the difference between **NONINSTANT*LOGVENDORS** and **INSTANT*LOGVENDORS**) is associated with a 1.4% decrease in mean time to patch vulnerabilities. After controlling for number of versions supported by the vendor, vendors release patch about 1.5% earlier when faced with a 10% higher quantity.

To put these results in perspective we interpret the elasticities in terms of number of days using sample mean values of vendors affected and competitors affected. With sample mean of number of vendors under *instant* disclosure being 6.22 when vulnerabilities are instantly disclosed, one vendor corresponds to a 16% increase in rivals faced by a vendor. Given that the mean of **DURATION** is about 168 days in the sample, the competition effect associated with one vendor, is about 8.33 days on an average. If a software vendor faces about 6.22 other vendors as competition on an average for a vulnerability, then, the average effect of competition is about 52 days. Similarly the presence of one **VENDOR** under *non-instant* disclosure corresponds to a 9.6% increase in rivals faced by a vendor. Using the sample mean value of **DURATION** the presence of one VENDOR decreases the mean time to release the patch by about 2.25days on an average due to disclosure threat If vendors on an average face about 10.43 (sample mean), the effect of disclosure threat is about 24 days on an average.

We are working on various specifications to understand the robustness of our estimates.

5. Discussion and conclusion:

The relationship between quality and market concentration has long been of interest to both policy makers and economists. In our application, we focus on the effect of competition on one aspect of software quality – time taken by software vendors to release patches that fix vulnerabilities. Using a unique dataset comprising of software vulnerabilities we examine if higher competition results in an early patch release by software vendors. Since patching vulnerabilities is similar to ex-post service quality implicitly we provide evidence of the effect of competition on quality. Overall, our results suggest that vendors respond to higher competition by patching vulnerabilities earlier. We also identify two different facets of competition namely disclosure threat, which is the threat of any one of the vulnerable vendors implicitly making a disclosure of vendor's quality and "competition effect" which is the effect of the presence of higher number of vulnerable rivals. Both, the possibility of earlier patch release by vulnerable vendors, which implicitly is also disclosure (disclosure threat effect), and, the fear of being penalized by end-users for late patch release relative to other vulnerable vendors (competition effect), result in vendors releasing patches early. Also, a larger market share induces vendors to release patches earlier. Given that empirical research on the impact of competition on quality has been sparse due to the complexity in measuring quality unambiguously, the evidence on the effect of competition on quality provided in this paper marks a contribution to the literature on how competition affects quality.

Our results also have implications on consumer welfare and vulnerability disclosure policies. If one were to believe that earlier patch release is highly correlated with lower end-user losses from vulnerabilities, our results suggest that higher competition, results in lower end user losses, thereby enhancing consumer welfare. Our results also show that disclosure threat can be used as a tool to induce vendors to patch vulnerabilities faster. Thus in part, the result of disclosure threat provides evidence that suggests that *non-instant* disclosure could be more welfare enhancing than *instant disclosure.* For policy makers like CERT/CC and security practitioners, this result provides valuable evidence to support *non-instant* disclosure, which uses disclosure threat rather than actual disclosure. Since the usefulness of disclosure policy is only to the extent of the threat of disclosure and its ability to make vendors respond faster to vulnerability designing an optimal disclosure policy would involve judiciously using disclosure threat to elicit proper vendor responses to vulnerabilities. Both of these findings are questions that have been empirically unanswered thus far in the economics of information security literature, to the best of our knowledge and hence also mark a contribution to the literature.

References:

Arora A., Caulkins J., Telang R. (2005) “Sell First, Fix Later: Impact of Patching on Software Quality", Management Science (Forthcoming)

Arora A., Krishnan R, Telang R. & Yang Y. (2005) "An Empirical Analysis of Vendor Response to Disclosure Policy," Workshop on Economics of Information Security (WEIS05), Kennedy School of Government, Harvard University, 2005.

Arora A., Nandkumar A. & Telang R. (2004) "Impact of patches and software vulnerability information on frequency of security attacks - An empirical analysis, Working paper," in: *H.John Heinz III school of Public Policy and Management*, Carnegie Mellon University, Pittsburgh, PA, 2004.

Arora A., Telang R. & Xu H. (2004)"Optimal Policy for Software Vulnerability Disclosure," The Third Annual Workshop on Economics and Information Security (WEIS04), University of Minnesota, 2004.

Borenstein S. and Netz J. (1999), “Why do All the Flights Leave at 8 am?: Competition and Departure-Time Differentiation in airline markets,” International Journal of Industrial Organization, 20(3):344-365

Cavusoglu H., H. Cavusoglu, S. Raghunathan (2005), "Recent Issues in Responsible Vulnerability Disclosure," Workshop on Economics and Information Security (WEIS), Boston, MA, June

Choi J.P., Fershtman C. & Gandal N. (2005) "Internet Security, Vulnerability Disclosure, and Software Provision," Workshop on Economics of Information Security (WEIS05), Kennedy School of Government, Harvard University, 2005.

Cohen A. and Mazzeo M. (2004) “Competition, Product Differentiation and Quality Provision: An Empirical Equilibrium Analysis of Bank Branching Decisions,” Finance and Economics Discussion Series 2004-46. Washington: Board of Governors of Federal Reserve System, 2004.

Dranove D. and W.White (1994), “Recent Theory and Evidence on Competition in Hospital Markets,” Journal of Economics and Management Strategy, 3(1):169-209.

Domberger S. and A. Sherr (1989), “The impact of competition on pricing and Quality of Legal Services,” International Review of Law and Economics, 9:41-56.

Forman C., GoldFarb A., and Greenstein S. (2005), “How did location affect adoption of the commercial
Internet? Global village vs. urban leadership” Journal of Urban Economics (Forthcoming)

Gal-Or E., (1983), “Quality and quantity competition” The Bell Journal of Economics, 14(2):590-600

Hoxby C. (2000), “Does Competition among Public Scools benefit Students or Taxpayers?,” American Economic Review, 90(5):1209-1238

Levhari D. and Peles Y., (1973), “Market Structure, Quality and Durability.” The Bell Journal of Economics and Management Science, 4(1): 235-248

Mazzeo M. (2003), “Competition and Service Quality in the U.S. Airline Industry,” Review of industrial Organization, 22: 275-296

Schmalensee R. (1979), “Market Structure, durability, and Quality: A Selective Survey,” Economic Inquiry, 17: 177-196

Schneier B. (2000) "Full Disclosure and the Window of Exposure," in: *CRYPTO-GRAM*, 2000.
Nizovtsev, D.T., M. "Economic Analysis of Incentives to Disclose Software Vulnerabilities," Workshop on Economics and Information Security (WEIS05), Kennedy School of Government, Harvard University, 2005.

Spence A.M., (1975), “Monopoly, Quality and Regulation” The Bell Journal of Economics 6(2): 417-429

Swan P.L., (1970), “Durability of Consumer Goods,” American Economic Review, 60: 884-894

Telang R. and Wattal S. (2005) "Impact of Software Vulnerability Announcements on the Market Value of Software Vendors – an Empirical Investigation," Workshop on Economics of Information Security (WEIS05), Kennedy School of Government, Harvard University, 2005.

Descriptive statistics

Variable	N	Proportion	Mean	Std. Dev
Proportion of Anti-Virus vulnerabilities	5	1.06	-	-
Proportion of Application Development vulnerabilities	3	0.63	-	-
Proportion of Application Server Software vulnerabilities	15	3.17	-	-
Proportion of Backup And Recovery vulnerabilities	1	0.21	-	-
Proportion of Data Base Management vulnerabilities	15	3.17	-	-
Proportion of Electronic Mail vulnerabilities	4	0.85	-	-
Proportion of Groupware Software vulnerabilities	12	2.54	-	-
Proportion of LAN Operating System vulnerabilities	2	0.42	-	-
Proportion of Operating System vulnerabilities	368	77.80	-	-
Proportion of Suites vulnerabilities	4	0.85	-	-
Proportion of System Utilities vulnerabilities	1	0.21	-	-
Proportion of System/Software Management vulnerabilities	1	0.21	-	-
Proportion of Web Browser vulnerabilities	26	5.50	-	-
Proportion of Web Design Tools vulnerabilities	1	0.21	-	-
Proportion of Web Development Tools vulnerabilities	4	0.85	-	-
Proportion of Web Server Software vulnerabilities	11	2.33	-	-
LOGQUANTITY	473	-	14.00	2.26
LOGDURATION	473	-	3.43	2.11
LOGDURATION - Instant disclosure	291	-	3.77	2.22
LOGDURATION - Non-Instant disclosure	182	-	2.87	1.80
# Vulnerabilities	241	-	-	-
# Vulnerabilities – Instant disclosure	183	-	-	-
# Vulnerabilities – Non- Instant disclosure	79	-	-	-
# Vulnerabilities – Non- Instant & instant	21	-	-	-
# observations for which vendor did not issue patch	20	-	-	-
# vulnerabilities for which vendor did not issue patch	15	-	-	-
Other vulnerable vendors	473	-	7.83	8.03
VENDORS – Instant disclosure	291	-	6.22	6.95
VENDORS - Non-Instant disclosure	182	-	10.43	8.93

SEVERITY	473	-	22.34	20.74
SEVERITY- Instant disclosure	291	-	22.09	20.57
SEVERITY- Non-Instant disclosure	182	-	22.71	19.74
# of distinct vendors	30			
Proportion Apple	26	5.50	-	-
Proportion Hewlett Packard	45	9.51	-	-
Proportion IBM (includes Lotus)	45	9.51	-	-
Proportion Microsoft	77	16.28	-	-
Proportion Oracle	22	4.65	-	-
Proportion SCO	55	11.62	-	-
Proportion SGI	22	4.65	-	-
Proportion SuSE	39	8.25	-	-
Proportion Sun Microsystems	43	9.09	-	-
Proportion Compaq	15	3.17	-	-
Proportion Redhat	60	12.68	-	-
Proportion of vulnerabilities identified by CERT	9	0.04	-	-
Proportion of vulnerabilities identified by University	10	0.04	-	-
Proportion of vulnerabilities identified by Consulting company	71	0.29	-	-
Proportion of vulnerabilities identified by end user	19	0.08	-	-
Proportion of vulnerabilities identified by Vendor	46	0.19	-	-
Proportion of vulnerabilities identified by individual	86	0.36	-	-