Information Economics and Policy 22 (2010) 164–177

Contents lists available at ScienceDirect

Information Economics and Policy

journal homepage: www.elsevier.com/locate/iep

Competition and patching of security vulnerabilities: An empirical analysis

Ashish Arora [a], Chris Forman [b], Anand Nandkumar [c,*], Rahul Telang [d]

[a] Fuqua School of Business, Duke University, 1 Towerview Drive, Durham, NC 27708, United States
[b] College of Management, Georgia Institute of Technology, 800 West Peachtree St. NW, Atlanta, GA 30308, United States
[c] Indian School of Business, Gachibowli, Hyderabad 500 032, India
[d] H. John Heinz III College, Carnegie Mellon University, 5000 Forbes Avenue, Pittsburgh, PA 15213, United States

ARTICLE INFO

Article history:
Received 21 June 2009
Received in revised form 16 October 2009
Accepted 27 October 2009
Available online 22 November 2009

JEL classification:
L10
L15
L86

Keywords:
Information security
Competition
Software quality
Vulnerabilities

ABSTRACT

We empirically estimate the effect of competition on vendor patching of software defects by exploiting variation in number of vendors that share a common flaw or common vulnerabilities. We distinguish between two effects: the direct competition effect when vendors in the same market share a vulnerability, and the indirect effect, which operates through non-rivals that operate in different markets but nonetheless share the same vulnerability. Using time to patch as our measure of quality, we find empirical support for both direct and indirect effects of competition. Our results show that ex-post product quality in software markets is not only conditioned by rivals that operate in the same product market, but by also non-rivals that share the same common flaw.

1. Introduction

Many, if not most, cyber attacks exploit software defects (see for example Arbaugh et al., 2000). It is widely believed that poor software quality is an outcome of the market power that software vendors enjoy. However, there is little, if any, empirical work that examines the relationship between software quality and the degree of competition. One of the main difficulties in undertaking such empirical work is the lack of variation in number of competitors. Almost all software product markets are national, if not global, making it difficult to estimate the effects of competition using regional variation in competition, as is commonly done for other industries. A second key challenge is to measure quality.

In this paper, we use two unique features of our data to overcome these challenges. First, we use the time taken by vendors to release a patch for a software vulnerability as our proxy for quality. Although all concede that it is better that a product not have bugs in the first instance, this is unrealistic and perhaps may even be too costly. Thus, patches are an important component of post-sales product support and timely patch release is an important part of overall information security (Arora et al., 2006a; Beattie et al., 2002).

Second, we use variation in the number of vendors affected by a common vulnerability (a vulnerability that affects products manufactured by different vendors, discussed in detail later) to empirically estimate the effects of competition. In particular, this variation enables us to examine how number of vendors affected by a vulnerability influences patch release behavior of software vendors. A timely patch is vital in limiting losses from cyber-attacks, which are increasing in the time elapsed between the initial disclosure of the vulnerability and the release of

* Corresponding author.
E-mail addresses: ashish.arora@duke.edu (A. Arora), chris.forman@mgt.gatech.edu (C. Forman), anand_nandkumar@isb.edu (A. Nandkumar), rtelang@andrew.cmu.edu (R. Telang).

0167-6245/$ - see front matter © 2009 Elsevier B.V. All rights reserved.
doi:10.1016/j.infoecopol.2009.10.002

the patch. Tardy patching likely reduces customers' willingness to pay for a vendor's current and future products. Therefore, how quickly a vendor will release the patch should depend upon how accurately customers are able to judge whether the vendor's patches were tardy or not, and also the choices available to the customer.

The degree of competition faced by the vendor affects both of these factors. For instance, in deciding how tardy a vendor is, customers may compare it to how quickly other vendors in the same market (henceforth rivals) provided a patch for the vulnerability. Thus, the number of rivals who are also developing a patch is one dimension of the degree of competition, and we label this the direct effect. Customers may also look at the patching performance of vendors affected by the same vulnerability but operating in other markets (henceforth non-rivals) in assessing the timeliness of the patch provided by their vendor. Thus, the number of non-rivals affected by the common vulnerability measures a different dimension of the degree of competition, which we label the indirect effect. Competition has a third dimension as well, which we call the disclosure effect. When a vendor releases a patch, it de facto discloses the vulnerability to all, and attackers can exploit all unpatched machines. This affects both rivals and non-rivals who are working to release their patches.

To test the relationship between competition and quality, we examine responses by 21 vendors to 241 vulnerabilities reported to CERT/CC from September 2000 to August 2003. Our results demonstrate that the direct, indirect, and disclosure effects play a significant role in shaping the speed with which vendors release patches to software.

Our research makes a significant contribution to our understanding of vendor's investment in software quality, in particular recent work in the information security literature that has examined vendor patch release behavior (Arora et al., forthcoming; Cavusoglu et al., 2005; Choi et al., 2005; Li and Rao, 2007). To our knowledge our research is the first to demonstrate how increases in disclosure threats from rivals and non-rivals influences investments in information security and software quality. Our paper also contributes by analyzing the relationship between competition and software quality. In particular, our research demonstrates that despite high levels of concentration in many software markets, the competition from vendors in other related markets works to reduce patch release times. This indirect form of competition can be as effective in reducing time to patch as increases in the number of direct competitors.

2. Related literature and contribution

This paper is related to four streams of research: economics of information security, software quality and software process, competition and quality provision, and competition in technologically related markets.

There is relatively little work that focuses on managerial or organizational issues in the information security domain. Only recently have researchers started investigating important economic questions in the area of information security. Our research is motivated by theoretical models of the relationship between the timing of vulnerability disclosure and the expected losses from attacks (Schneier, 2000; Arora et al., 2008; Cavusoglu et al., 2005) and more broadly research that has studied the factors shaping the timing and nature (public or private) of vulnerability disclosure by firms and third parties (Kannan and Telang, 2005; Nizovtsev and Thursby, 2007; Choi et al., 2005).

More recently, researchers have also focused on understanding the empirical implications of computer security. For example, Tucker and Miller (2008) show that concerns related to privacy and information security can inhibit diffusion of networked IT. Hann et al. (2007) evaluate the effectiveness of various online privacy policies using an information processing theory approach. Recently some empirical work has examined the economic implications of vulnerability disclosure. Arora et al. (2006b) find that disclosure of information about vulnerabilities increases frequency of attacks, especially if the patch is not available. Even the release of a patch results in a temporary increase in attacks but a sharp decline thereafter, resulting in a lower average attack frequency. Arora et al. (2008) use a dataset assembled from CERT/CC's vulnerability notes and SecurityFocus database to show that early disclosure leads to faster patch release times. Telang and Wattal (2007) use an event study methodology to show that vulnerability disclosure leads to a loss of market value. Li and Rao (2007) empirically examined the role of private intermediaries on the timing of patch release by vendors and found that the presence of private intermediaries decreases vendors' incentive to deliver timely patches. Our research is similar to prior work in that we examine the economic outcomes from vulnerability disclosure. However, in contrast to the prior work in this area, we study the relationship between competition and vendor patch release times.

The software community has long been concerned with the determinants of software quality. The literature has examined the link between quality and software development process (e.g., Banker et al., 1998; Harter et al., 2000; Agarwal and Chari, 2007). These studies conclude that a higher level of software process maturity is associated with better software. Our study is different from this prior work in two important aspects: First, we focus on ex-post quality rather than pre-release software quality. Second, in an advance over the literature, we explicitly examine the link between software quality and competition.

While a rich theory literature has examined the link between competition and quality, empirical work has been limited due to the inherent challenges of measuring product quality.[1] In general, prior work has demonstrated that increases in competition lead to better quality provision (e.g., Domberger and Sherr, 1989; Dranove and White, 1994; Borenstein and Netz, 1999; Hoxby, 2000; Mazzeo, 2003; Cohen and Mazzeo, 2004). However, most prior work in this literature has focused upon services industries such as banking, legal or health services in which markets are local and empirical estimates are identified using cross sectional

[1] Prior theory work has demonstrated that increases in concentration can lead to an increase or decrease in product quality. For examples, see Gal-Or (1983), Levhari and Peles (1973), Schmalensee (1979), Swan (1970), and Spence (1975).

variation across geographic markets. In contrast, we examine this relationship within the context of a major product market, software, and obtain identification using variation in the number of products affected by software vulnerabilities.

While prior work has demonstrated a link between competition and product quality, it has not studied the interaction between firms in technologically related markets as we do. Recent work has highlighted the impact of firm strategic decisions in technologically related markets (e.g., Bresnahan and Greenstein, 1999; Bresnahan and Yin, 2006; Kretschmer, 2005; West and Dedrick, 2000). However, this research has focused on markets that are complements in demand. We argue that vendors who share common inputs will have important implications for vendors' quality decisions. To our knowledge, ours is one of the first papers to demonstrate empirically the interrelationships of strategic decisions among firms that share common inputs. Such interrelationships are likely to be particularly salient in software markets, where vendors in different market segments increasingly share common modules (e.g., Banker and Kauffman, 1991; Brown and Booch, 2002).

3. Conceptual framework and hypotheses

Unlike defects in physical goods, software defects can be mitigated even after product release via patch release (Arora et al., 2006a). This makes both vulnerabilities in software, as well as patches that fix vulnerabilities, common among software products. The probability of a malicious attacker exploiting a specific vulnerability to compromise end user computers is positively related to the time the vulnerability remains without a fix. Thus, the timing of patches critically determines the extent of end user losses, and patches are perceived as a very important part of ex-post customer support.

We focus here on two considerations that drive the timing of a vendor's patch: the cost of developing the patch and the extent of user losses that the vendor internalizes.[2] Typically, an early patch entails higher costs but also lower customer losses. In turn, the customer losses internalized by a vendor depend on the extent of market competition and the number of end users (or market size). The first factor that influences the timeliness of patch release is the degree of competition. Prior literature from other industries has shown that increases in competition are associated with greater product quality due to firm efforts to vertically differentiate themselves to win customers over from their competitors (Domberger and Sherr, 1989; Dranove and White, 1994; Mazzeo, 2003; Cohen and Mazzeo, 2004). In our research, we measure the effects of competition on quality by examining how increases in the number of other firms affected by a vulnerability influence patching times. Greater competition, especially from rivals, implies that end users are more likely to penalize lagging vendors because users have more alternatives, and because the lagging vendor appears less responsive than others that patch more quickly, and thus suffers a bigger blow to its reputation.

Hypothesis 1. Vendors that face more rivals affected by the same vulnerability are more likely to release a quicker patch.

In many cases, a newly discovered vulnerability could affect many different products (for future reference we label these common vulnerabilities). A common vulnerability is typically due to a shared code base or design specification, or due to a proprietary extension of a widely used software component. An example is a stack buffer overflow vulnerability in Sendmail (a commonly used mail transfer utility),[3] disclosed in 2003, that affected the following vendors: Apple, Conectiva, Debian, FreeBSD, Fujitsu, Gentoo Linux, Hewlett–Packard, IBM, MandrakeSoft, Mirapoint, NetBSD, Nortel Networks, OpenBSD, OpenPKG, Red Hat, SCO, Sendmail Inc., Sequent (IBM), SGI, Slackware, Sun Microsystems, SuSE, The Sendmail Consortium, Wind River Systems, and Wirex. Some of the products produced by these vendors potentially compete with one another while others are in very distinct markets. For example, Wirex and Mirapoint produce email products, Wind River produces embedded software, while many of the other products are operating systems. Even among the latter, there is considerable variation in the hardware platforms used. However, all these products use Sendmail code, and hence were affected by the vulnerability.

When a vulnerability is common to many products, customers can compare how quickly the vendor releases the patch relative to vendors affected by the vulnerability but operating in different markets (non-rivals henceforth; we will refer to such competition as "indirect" competition). Thus the number of vendors operating in a different market is also likely to influence the timing of patch release by a vendor. Although the literature on competition and product quality has often stressed the ability of consumers to compare among different firms, our setting is unique in that comparisons may occur among firms in different market segments as well.

Hypothesis 2. Vendors that face a larger number of indirect competitors affected by the same vulnerability are more likely to release a quicker patch.

In addition to letting customers judge more precisely whether their vendor is releasing patches in a timely manner, the number of other firms affected by the same vulnerability affects patching behavior through another route that we label the disclosure effect. Users' expected losses from software vulnerabilities will be higher when these vulnerabilities have been publicly disclosed pending a patch release, because public disclosure makes it easier for attackers to find vulnerabilities (Arbaugh et al., 2000; Arora et al., 2006b). Because software vendors internalize some fraction of users' losses, vulnerability disclosure is

[2] Yet another consideration is whether or not there is likely to be a version release in the near future – in such cases the vendor whose product is affected by the vulnerability may instead prefer to accelerate the release of newer versions rather than providing a patch for the vulnerability that is deployed on the older version. In this paper we focus on the cost of early patch development traded off against user losses internalized by the vendor.

[3] Vulnerability number VU#897604 by CERT/CC classification. See http://www.kb.cert.org/vuls/id/897604 (accessed 09/22/2006).

associated with briefer times to patch release (Arora et al., 2006). Prior work has focused on how public disclosure of vulnerabilities by third party intermediaries shapes patch release behavior (Schneier, 2000; Arora et al., 2008; Cavusoglu et al., 2005). While disclosure of vulnerabilities by third parties is important, a vulnerability can also be disclosed when someone issues a patch for it.

In deciding how expeditiously to develop and release a patch, a firm must consider how quickly the vulnerability is likely to be disclosed. The greater the number of firms (rivals or non-rivals) affected by the vulnerability, the more likely that someone will release a patch disclosing the vulnerability. In other words, the greater the number of vendors affected by a vulnerability, the greater the threat of disclosure, and hence, the more expeditiously each vendor will try to develop and release a patch.

Thus, increases in the number of affected vendors will also influence patch release times indirectly through disclosure. We label this the disclosure effect. Below, we describe in detail how this is identified separately from the effects of direct and indirect competition.

Hypothesis 3. Vendors facing a greater threat of disclosure are likely to release a patch sooner.

In focusing on these three dimensions of the degree of competition, we are mindful that we are neglecting the most obvious dimension, namely the total number of sellers in the market, whether or not they are affected by a given vulnerability. When there are many competing products, end users have more choices, and thus, future sales of a product may be more sensitive to perceived quality (Levhari and Peles, 1973; Spence, 1975; Schmalensee, 1979). This apparent neglect of the total number of sellers reflects the nature of our data – there is no variation in the total number of sellers within a market. We account for this by using product market dummies in our empirical specifications.[4] However, the direct effect of competition identified in this paper is a more restricted mechanism, namely how differences in number of rivals affected by a vulnerability influence the speed with which a patch is released for that vulnerability.

A second factor that determines the total customer losses incurred due to the vulnerability is the number of end users (or market size). Roughly speaking, a greater number of end users increases the total losses from the vulnerability and may also increase the attractiveness of the vulnerability to a malicious attacker. In general, attackers prefer exploiting popular products relative to obscure ones. Thus the likelihood that the vulnerability is exploited by an attacker is likely to be higher for popular products (Honeynet Project, 2004; Symantec, 2004). Moreover, the sheer number of end users also implies greater monetary losses that a vendor internalizes from a vulnerability. While other research has explored how vendor size influences the speed with which vendors release patches (Arora et al., 2006), to our knowledge we are the first to investigate how market size influences time to patch release.

Hypothesis 4. Vendors with larger market size are likely to release a patch sooner.

4. Data and variables

We assembled our data set of vulnerabilities from notes published by CERT/CC.[5] The vulnerabilities analyzed in this study were published by CERT between September 2000 and August 2003. On average, about 3000 vulnerabilities are reported to CERT/CC in a year, of which only about 10% are deemed legitimate and significant enough to be published. After determining if a reported vulnerability is authentic and exceeds CERT/CC's minimum threshold value for severity, as measured by the CERT METRIC (described later), CERT/CC's staff contact vendors that in its view may be affected by the vulnerability. CERT tends to contact as many vendors as possible even those who it suspects may be remotely affected by the vulnerability. Vendors then respond back to CERT whether they are vulnerable or not and in many cases with a list of products that are affected by the vulnerability. A vendor's response can typically be one of the following. The vendor may acknowledge the vulnerability in its product(s). In this case, CERT/CC lists the product's status as "vulnerable." The vendor may report that the product is not vulnerable, in which case CERT/CC lists the vendor's status as "not vulnerable." The vendor may also choose not to respond: In this case, CERT/CC records the vendor's status as "unknown."

Our unit of observation is a vendor–vulnerability pair. Our goal is to estimate the influence of competition on how long an affected vendor takes to provide a fix for the vulnerability. Given CERT's strategy of contacting many vendors, even those who may not be affected, we only considered vendors that acknowledged that their product(s) were vulnerable as those that were affected by the focal vulnerability. It is nonetheless quite plausible that a vendor might in fact be affected by the focal vulnerability even when CERT lists the status for a vendor–vulnerability pair as "unknown." However, we have no practical way of ensuring of whether the vendor was actually affected or not in such cases. Hence, the resulting bias, if any, cannot be determined.[6] Our sample consists of 1714 vendor–vulnerability pairs that were listed as "vulnerable" by CERT/CC. From this set, we dropped observations that relate to non-commercial entities (such as universities and not-for-profit vendors) and foreign vendors (vendors that do not have significant sales in the US).[7] Non-commercial

[4] As a robustness check, we also re-estimated our results using a single market (operating systems), with very similar results, indicating that any potential bias is very small.

[5] Other data sources such as online forums do not usually give a "protected period" to vendors to patch vulnerabilities before disclosing them publicly. Also, other sources also do not verify vulnerabilities in the same way that CERT does.

[6] Arora et al. (2006) suggest that these many of the vulnerabilities that were not acknowledged by vendors may not be genuine. However, in cases where the vulnerability may have been genuine, but were not acknowledged by the focal vendor, the focal vendor was unlikely to fix the vulnerability.

[7] The list of eliminated vendors and non-commercial entities consists of Apache, BSD (FreeBSD, OpenBSD), Debian, GNU, Gentoo, ISC, KDE, MIT Kerberos, OpenAFS.org, OpenLDAP project group, OpenBSD that makes OpenSSH, OpenSSL project group, Openwall GNU Linux group, Samba Team, Sendmail Inc., Slackware, Sorceror Linux, Stunnel, Tcpdump.Org, The Linux Kernel Archives, Trustix, University of Washington, XFree86, Xpdf, Yellow Dog Linux, mod ssl and zlib.org.

entities may have other objectives other than just maximizing profits and hence may not be subjected to the same pressures as for-profit vendors. Moreover, we are also unable to measure market size for non-commercial entities or foreign vendors reliably.[8] Many of the non-commercial entities have very few corporate customers, which, as we will explain later, is our measure of quantity. Also many of the foreign vendors typically have customer bases that mainly consist of non-US customers, while our measure of quantity is based on US corporate customers. We hence do not include observations that relate to such vendors in our empirical analysis. However, as we will explain in detail later, we include both non-commercial as well as foreign vendors in our measures of competition.

We also removed protocol vulnerabilities from the sample, as patches to these vulnerabilities typically involve protocol changes whose scope extends beyond a particular product. In many cases, even if a vendor knew the existence of such vulnerabilities, fixing it involves changes not just to the product but also the underlying protocol, which may involve cooperation of other involved parties. Protocol vulnerabilities thus typically do not conform to the phenomenon considered in this paper. Finally, we dropped observations wherein the vendors discovered and disclosed the vulnerability to CERT/CC of its own accord along with a patch. In such cases, since we cannot reliably measure when the vendor came to know of the existence of the vulnerability, we cannot also reliably determine how long the focal vendor took to release a patch. Our final sample includes 241 distinct vulnerabilities and 461 observations.[9]

We use variance in the manner with which vulnerabilities are disclosed to identify the competition and disclosure effects. From CERT/CC data (and discussions with CERT/CC staff), we know the date when a vendor is notified of the vulnerability. CERT/CC also records if and when the vulnerability was publicly disclosed. Thus, we label vulnerabilities as instantly disclosed if the existence of the vulnerability had been publicly disclosed (by some third party) prior to CERT/CC's notification to the vendor. We label vulnerabilities as non-instantly disclosed when CERT/CC discloses a vulnerability that had previously not been publicly disclosed.

4.1. Dependent variable

Our dependent variable is DURATION, a measure of the number of days a vendor takes to release the patch. Measurement of DURATION depends on the regime of disclosure – instant or non-instant. If the vulnerability is instantly disclosed, DURATION is the elapsed time in days between the date when the vulnerability was publicly disclosed and the date when the vendor released the patch. If the vulnerability is non-instantly disclosed, DURATION is the elapsed time between the CERT/CC notification to the vendor and the date when the vendor released the patch. For the empirical analysis we use the log of (1 + DURATION) as our dependent variable. We label this variable LOGDURATION. Of the 461 observations in our sample, 4.3%, or about 20 observations, had no patch. For these unpatched observations, we assign the maximum value of LOGDURATION that we observed in our sample (8.27). As we will show, our results are unchanged when we use a Tobit model that treats these observations as right-censored. Table 2 provides the descriptive statistics for LOGDURATION.

Table 1
Variable descriptions.

Variable	Description
DURATION	Time taken by vendors to issue a patch for a vulnerability
LOGDURATION	Log of DURATION
VENDOR	Total number of vulnerable vendors affected
RIVAL	Number of vulnerable sellers in the same market
NON-RIVAL	Number of vulnerable sellers in other markets
INSTANT	1 if instant disclosure, 0 otherwise
NON-INSTANT	1 if non-instant disclosure, 0 otherwise
LOGQUANTITY	Log(1 + total # of employees at customer sites (sites that use the software))
LOGVERSIONS	Log of number of versions
LOGSEVERITY	Log(1 + CERT severity metric)
SCORE	Vulnerability severity score (CVSS base score)

Table 2
Descriptive statistics.

Variable	Full sample $N = 461$			
	Mean	Minimum	Maximum	Standard deviation
DURATION (days)	168	1	3904	558
LOGDURATION	3.52	0.69	8.27	1.92
VENDORS	9.02	1	37	8.04
RIVALS	5.96	0	19	5.87
NON-RIVALS	3.03	0	24	3.65
LOGQUANTITY	13.95	6.22	17.41	2.26
LOGVERSIONS	0.22	0	3.14	1.63
LOGSEVERITY	2.73	0	4.69	20.34
LOGSCORE ($N = 187$)	1.86	0.18	2.30	0.51

4.2. Independent variables

A description of all independent variables is included in Table 1, while descriptive statistics are included in Table 2.

Competition: to measure how the different dimensions of the degree of competition influence patch release times, we construct three variables. RIVALS is the number of vendors that CERT lists as vulnerable and that operate in the same product market. NON-RIVALS is the number of vendors that are vulnerable but operate in a different market. We determined rivals and non-rivals using market definitions in the Harte–Hanks CI Technology database

[8] Foreign vendors include Mandrake Linux that is headquartered in France and Turbo Linux, headquartered in Japan. The results are qualitatively unchanged even if we include foreign vendors.

[9] An analysis of mean differences of the number of rivals and non-rivals between the sample that consists of vendors retained for empirical analysis and the sample of vendors excluded suggests that the means of key variables are not statistically different from each other. This suggests that it is unlikely that dropping observations based on the criteria outlined above introduces any systematic selection biases to our empirical estimates.

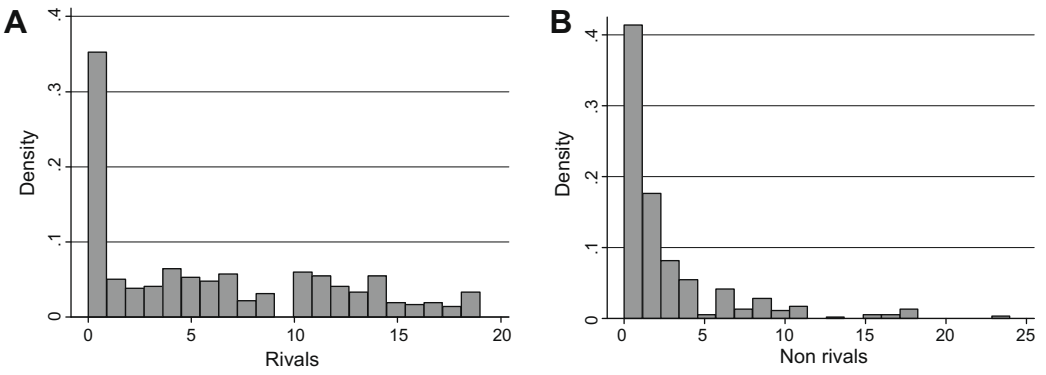

Fig. 1. (A) and (B) Histograms of RIVALS and NON-RIVALS.

(hereafter CI database).[10] As an example, suppose the focal vendor–vulnerability pair was Microsoft-Windows XP vulnerability and the vulnerability was shared by products produced by Red Hat and Oracle. In this case, RIVALS consists of Red Hat (since both Red Hat and Microsoft are in the operating system market). NON-RIVALS consist of Oracle, while VENDORS consists of both Red Hat and Oracle. As explained earlier, although we exclude observations that relate to non-commercial entities and foreign vendors, we include both groups in our measures of competition. As a robustness check, we have re-estimated our regressions using measures of competition excluding non-commercial and foreign vendors and the results are qualitatively similar. In Fig. 1A and B we produce the histograms of RIVALS and NON-RIVALS. The figures show that a majority of vulnerabilities in the sample affect more than one vendor – about 65% of the vulnerabilities have more than one RIVAL and about 60% of the vulnerabilities have more than one NON-RIVAL.

Quantity: Data on cumulative sales quantity for a product was collected using 2002 data from the CI database. The database reports only binary decisions of software use in a firm or establishment: details on the number of copies of a software product are not reported. To develop a measure of the total installed base of a software product, we use the number of firms that indicated use of the product and weight it by the number of employees in the organization. For instance if 1000 establishments own at least one 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 firms. This puts more weight on products used in larger firms, and arguably provides us a more accurate proxy for quantity. Finally, we follow Forman et al. (2005) and weight our data using County Business Patterns data from the US Census to correct for oversampling of some industry sectors in the CI database. In sum, to compute our final measure of quantity, we multiply the binary measure of software use for each firm by the number of firm employees and by firm weights. We then sum across firms. As the distribution of quantity is highly skewed, we take the log of quantity (LOGQUANTITY) for our analysis.

Other variables: In order to account for differences in severity of vulnerabilities we use the log of (one plus) CERT's severity metric, which we label LOGSEVERITY. The CERT severity metric is a number between 0 and 180, and is a comprehensive measure of the severity of vulnerabilities. CERT provides this metric to vendors as a guide to enable them to identify the serious vulnerabilities from the larger number of less severe vulnerabilities. We use this as our principal control for unobserved vulnerability characteristics. CERT uses an extensive set of criteria, including whether (i) information about the vulnerability is widely available; (ii) the vulnerability is being exploited in incidents reported to US-CERT; (iii) the Internet infrastructure is at risk because of this vulnerability; (iv) large number of systems on the Internet that are at risk from this vulnerability; (v) the impact on users of an exploit is high; and (vi) the vulnerability can be easily exploited, e.g., whether it can be exploited remotely or not.[11] CERT's criteria do not explicitly take into account the number of vendors affected by the vulnerability. To guard against the possibility that unobserved features of the vulnerability that drive patching speed are correlated with the number of affected vendors, we performed several robustness checks.

One such check involves using an alternative summary measure of the characteristics of vulnerabilities, the Common Vulnerability Scoring System (CVSS) base score,[12] a measure developed by the National Infrastructure Advisory Council (NIAC). The NIAC is an industry group which provides the Department of Homeland Security with recommendations for IT security of critical infrastructure. The CVSS base score is a numeric score ranging from 0 to 10, that represents the intrinsic qualities of a vulnerability. This measure is correlated with the CERT metric (correlation = 0.57), but not perfectly. Since this scoring system was launched only in 2005, this score is only available for only 96 vulnerabilities in our sample. We re-estimated our specification for this sub-sample, using the natural log of the CVSS score, LOGSCORE, as a measure of the severity of

[10] In those cases where the product was not included in the database, we examined product manuals to classify the product.

[11] See www.kb.cert.org/vuls/html/fieldhelp (last accessed on January 12, 2007).

[12] See http://www.first.org/cvss/cvss-guide.html (last accessed September 9, 2009).

the vulnerability.[13] We also performed a vulnerability fixed effect regression for a sample using the more widespread vulnerabilities, where we exploit variation in the number of non-profit and foreign vendors affected. In this way, we guard against the possibility that the (unobserved) ease of patching is correlated with how widespread the shared code is.[14]

Anecdotal evidence from industry sources suggests that quality testing of patches on multiple versions consumes additional time in the patch development process. Thus, we also control for the log of the number of software versions that have been produced (LOGVERSIONS). Descriptive statistics for all of the independent variables are included in Table 2.

5. Empirical models and results

In this section, we describe our method for identifying how competition and disclosure influence vendors' patch release times. We also discuss the results of our baseline empirical analysis.

5.1. Empirical model

Our goal is to examine how our proxy for ex-post quality – namely the duration of patch release time for vendor i in market m facing vulnerability v – varies with changes in competition. If $DIRECTCOMP_{iv}$ represents the effects of direct competition (competition from rivals), $INDIRECTCOMP_{iv}$ represents that of indirect competition (competition from non-rivals), while $DISCLOSURE_{iv}$ represents the effects of increased disclosure arising from a greater number of affected vendors, one may estimate the following linear model[15]:

$$\begin{aligned} LOGDURATION_{imv} = {} & \beta_0 + \beta_1 DIRECTCOMP_{iv} \\ & + \beta_2 INDIRECTCOMP_{iv} \\ & + \beta_3 DISCLOSURE_{mv} \\ & + \beta_4 LOGQUANTITY_{im} + \theta_1 X_i \\ & + \theta_2 Z_v + \theta_3 K_m + \varepsilon_{imv} \end{aligned} \quad (1)$$

where X_i is a vector of vendor characteristics that include vendor fixed effects for large vendors, K_m a vector of market fixed effects and Z_v is a vector of vulnerability characteristics that includes the severity metric. Our interest is in identifying the parameters β_1 through β_4 which correspond to hypotheses 1–4 and reflect the effects of direct competition, indirect competition, disclosure, and market scale, respectively.

Competition enables users to compare and benchmark the performance of their vendor relative to others that share the same vulnerability. These effects, captured by β_1 and β_2 in Eq. (1), could arise either from rivals or non-rivals. Competition also allows users to switch to a vendor providing a superior mix of price and quality. This type of competition can only be provided by other sellers in the same product market, whether or not they are share the vulnerability.

We use market fixed effects to control for unobserved factors that vary across markets. Our model includes dummies for the three largest markets, which account for 88% of the sample.[16] A small percentage (12%) of observations is from small markets that have insufficient observations to identify an individual market dummy. However, our results are robust to their exclusion. We also include firm dummies for the eight leading vendors, who jointly account for about 85% of the observations in our sample.[17] Estimates using only the top eight vendors with a full set of vendor fixed effects yield results similar to those reported.

We assume that LOGQUANTITY is statistically exogenous. In support of this assumption we note that LOGQUANTITY reflects the stock of installations in the CI database in 2002, rather than the purchase quantity in any particular year. However, LOGQUANTITY may reflect recent demand, which may be correlated with unobservable factors that influence patch release times. If so, our estimates would overstate the relationship between cumulative sales and quality provision, and potentially bias other estimates as well. However, excluding LOGQUANTITY (not shown here) yields very similar estimates for other variables, indicating that the bias, if any, does not extend to other variables of interest.

The effect of LOGQUANTITY on patch release times may be different for software vendors that also sell hardware: Such firms may also internalize the effect of vulnerable software on related hardware sales. For example, vulnerabilities in Sun's Solaris operating system may influence sales of its workstations too, shifting the relationship between installed base of Solaris and patch release times compared to other software firms. Conversely, if a vendor's main source of revenue is from hardware sales, such vendors may be less sensitive to software defects. To capture these potential differences, we interact LOGQUANTITY with a vendor hardware dummy that is equal to one when a software vendor also sells hardware (HARDWARE).[18] Re-estimation of models without including HARDWARE and HARDWARE $*$ LOGQUANTITY as covariates yields similar

[13] Correlation between LOGSEVERITY and LOGSCORE is 0.57.

[14] Moreover, a comparison of the correlations between LOGDURATION with RIVALS and NON-RIVALS when LOGSEVERITY was "low" (below sample median og LOGSEVERITY) and when LOGSEVERITY was "high" are similar suggesting that the unobserved heterogeneity if any, is unlikely to influence the number of RIVALS and NON-RIVALS for a vulnerability.

[15] A duration model is equally well suited for the underlying problem considered in this paper. In fact, a Cox specification that estimates Eq. (2) yields qualitatively similar results. We preferred a linear model to a duration model because we neither have time varying covariates nor any conditioning events (e.g., how many rivals have already disclosed), so that a hazard model and linear regression are likely to provide qualitatively similar estimates. Also, methods that do not rely on maximum likelihood, such as regression, are somewhat more robust and less sensitive to the specification of the error term.

[16] These include dummies for the operating system, application server and web browser markets.

[17] These are Apple, HP (includes HP, Compaq, and Digital), Microsoft, Sun, SCO, Red Hat, IBM (includes Lotus, iPlanet) and Oracle. The omitted category consists of smaller vendors with few observations: Adobe, SGI, Allaire, Macromedia, Netscape, Network Associates, Novell, Symantec, Trend Micro, and Veritas.

[18] In the dataset the hardware vendors are HP (including Compaq and Digital), Sun Microsystems and IBM.

Table 3
Comparison of conditional mean of LOGDURATION.

RIVALS	HIGH	LOW	COMPETITION EFFECT
LOGDURATION	2.81***	3.73***	−0.92***
	(0.21)	(0.12)	(0.24)

Notes: cells contain mean of LOGDURATION conditional on RIVALS. Standard errors in parentheses. Sample mean of RIVALS = 5.96.
*Significant at 90% confidence level.
**Significant at 95% confidence level.
*** Significance at 99% confidence level.

estimates of β_1 and β_2, although it yields different estimates of β_4.

Since it is plausible that errors in our sample may be heteroskedastic, we estimate a random effects GLS specification.[19]

5.2. Baseline empirical model: Identifying the effects of rivals only

We use several approaches to identify the coefficients β_1 through β_4 to improve confidence in our estimates. We begin with a simple comparison of sample means and then proceed with discussing the results of the regressions.

In Table 3 we provide some preliminary evidence on the effects of competition through an examination of conditional means. We categorize RIVALS as "high" if the number of affected RIVALS (direct competitors) for a vulnerability is above the mean and "low" otherwise. An increase in the number of RIVALS from below the mean to above the mean lowers LOGDURATION by a statistically significant 0.92.

Next we present regression results where we simply estimate how variation in the number of affected rivals affects patching time. Thus, we only capture only the effect of direct competition

$$LOGDURATION_{imv} = \beta_0 + \beta_1 RIVALS_{iv} + \beta_4 LOGQUANTITY_{im} + \theta_1 X_i + \theta_2 Z_v + \theta_3 K_m + \varepsilon_{imv} \quad (2)$$

We estimated Eq. (2) using OLS. The Breusch–Pagan test overwhelmingly rejects the assumption of homoskedasticity (χ^2 145.68; p-value = 0.00) so henceforth we use random effects models for our baseline linear estimates.

In Table 4 we present three sets of estimates. In column (1), we estimate Eq. (2) using a sub-sample comprising of observations from the operating system market (OS sample henceforth). In columns (2) and (3) we present estimates of the full sample with and without market dummies. These results suggest that an increase in the number of rivals decreases patch release times by about 7–9% or between 12 and 15 days per rival. Comparing between column (2) and column (3), the estimated effect is stable to the addition of market dummies. Quantity also decreases patching times: A 10% increase in quantity is associated with about a 1.3% decrease in patch release times or about 2 days. Thus, our analysis supports hypotheses 1 and 4. Interestingly, the coefficient of HARDWARE * LOGQUANTITY is positive and significant while the coefficient for HARDWARE is negative and significant in the full sample. This suggests that while hardware vendors on an average release patches earlier, they are not sensitive as pure software producers to their market size.

The results in columns (2) and (3) are similar to (1), where we only use the data from the operating system (OS) market. In part, this reflects the dominance of operating systems vulnerabilities in our sample. In addition, it also implies that that the possible unobserved differences across product markets in ease of patching and number of producers is not a major source of concern.

In column (4) we re-estimate Eq. (2) using vulnerability fixed effects, for the sub-sample of vulnerabilities that affected at least five rivals. This leaves us with only 162 observations comprising 31 vulnerabilities. This is the most stringent control for unobserved differences across vulnerabilities in terms of ease of patching and number of affected firms, because we also include market and vendor fixed effects. Although the effects of β_1 and β_4 are not precisely estimated, the direction, as well as the magnitude, of the point estimates is very similar to those in other specifications. This suggests that unobserved heterogeneity is unlikely to drive our results.

In column (5) we include LOGSCORE instead of LOGSEVERITY using the set of vulnerabilities for which SCORE was available (this comprises 187 observations relating to 96 vulnerabilities). Once again our estimates of the effect of direct competition (β_1) are similar to those of column (3). However, the estimated effect of LOGQUANTITY, HARDWARE and HARDWARE * LOGQUANTITY appear to be higher in magnitude, although directionally similar.[20]

5.3. Identification using variation in rivals, non-rivals and disclosure

The specification laid out in Eq. (2) ignores non-rivals, i.e., indirect competitors. It also ignores the disclosure threat that might arise from both rivals and non-rivals. In this section, we expand the specification to explore both of these additional dimensions of competition.

One challenge we face is to separately identify the direct and indirect effects of competition from the disclosure effect. The answer lies in another source of variation in our data. The conceptual framework outlined in Section 3 has the following implications: the threat of disclosure arises from increases in the number of rivals and non-rivals affected by the same vulnerability and arises only under non-instant disclosure. Put another way, when many

[19] We test for, and are unable to reject, the presence of unobserved heterogeneity.

[20] We also estimated additional specifications that also yielded qualitatively similar results (not reported here). We check if large vendors are less likely to be sensitive to the number of end users or to the number of RIVALS by including a dummy variable = 1 (called LARGE VENDOR) if the focal vendor had greater than 40% market share in a software market along with interaction of this variable with LOGQUANTITY and RIVALS. While the coefficients of LOGQUANTITY and RIVALS were statistically similar those of column (3) of Table 4, we did not find large vendors any more or less likely to fix vulnerabilities faster relative to smaller vendors. Furthermore, we also did not find any significant differences between smaller and larger vendors with regard to their sensitivity to the number of end users.

Table 4
Estimates of random effects GLS regressions of Eq. (2) – dependent variable LOGDURATION.

Variable	OS sample (1)	Full sample (2)	Full sample, market fixed effects (3)	Rivals > 5 vulnerability and market fixed effects (4)	CVSS metric, market fixed effects (5)
RIVALS (β_1)	−0.09***	−0.09***	−0.07***	−0.08	−0.11**
	(0.03)	(0.03)	(0.03)	(0.06)	(0.05)
LOGQUANTITY (β_4)	−0.03	−0.14**	−0.13*	−0.09	−0.25***
	(0.17)	(0.06)	(0.08)	(0.07)	(0.10)
HARDWARE	−2.75	−3.90**	−4.20***	−3.14	−11.27***
	(3.74)	(1.66)	(1.69)	(3.89)	(2.99)
HARDWARE * LOGQUANTITY	0.21	0.29***	0.24	0.24***	0.83***
	(0.25)	(0.11)	(0.31)	(0.10)	(0.25)
LOGVERSIONS	0.18	0.22	0.24	0.12	0.47*
	(0.21)	(0.17)	(0.17)	(0.37)	(0.28)
LOGSEVERITY	−0.14	−0.13	−0.14		
	(0.17)	(0.13)	(0.13)		
LOGSCORE					−0.79*
					(0.49)
Constant	6.15***	6.52***	7.07***	5.13	9.06***
	(0.88)	(1.01)	(1.08)	(4.56)	(1.42)
N	366	461	461	162	187
R^2 (between)	0.12	0.11	0.13	0.09	0.19
R^2 (within)	0.06	0.06	0.06	0.08	0.09
R^2 (overall)	0.12	0.14	0.15	0.09	0.18
# vulnerabilities	159	241	241	31	96
Market fixed effects	No	No	3	2	3
Vendor fixed effects	7[a]	8	8	4	8
σ_u	1.71	1.73	1.71	1.45	1.69

Notes: standard errors in parenthesis.
* Significant at 90% level.
** Significant at 95% level.
*** Significance at 99% level.
[a] Oracle is not an OS vendor.

vendors are affected by the vulnerability and the vulnerability has not yet been publicly disclosed, affected vendors face a disclosure threat (over and above the direct and indirect competition threat). In contrast, when the vulnerability is already disclosed (instant disclosure), vendors begin to develop patches knowing that attackers are also aware of the vulnerability, the direct and indirect competition effects operate but the disclosure threat is absent. Since some vulnerabilities in the sample are instantly disclosed and others are not, this provides the basis for the identification strategy shown below:

$$\begin{aligned} LOGDURATION_{imv} = {} & \beta_0 + \beta_1 RIVALS_{iv} \\ & + \beta_2 NON\text{-}RIVALS_{iv} \\ & + \beta_3 (1 - INSTANT_v) \\ & * (RIVALS_{iv} + NON\text{-}RIVALS_{iv}) \\ & + \beta_4 LOGQUANTITY_{im} \\ & + \theta_1 X_i + \theta_2 Z_v + \theta_3 K_m + \varepsilon_{imv} \end{aligned} \quad (3)$$

Collecting terms gives us the following estimating equation:

$$\begin{aligned} LOGDURATION_{imv} = {} & \beta_0 + \gamma_1 RIVALS_{iv} \\ & + \gamma_2 NON\text{-}RIVALS_{iv} + \gamma_3 INSTANT_v \\ & * (RIVALS_{iv} + NON\text{-}RIVALS_{iv}) \\ & + \beta_3 INSTANT_v \\ & + \beta_4 LOGQUANTITY_{im} + \theta_1 X_i \\ & + \theta_2 Z_v + \theta_3 K_m + \varepsilon_{imv} \end{aligned} \quad (4)$$

where $\gamma_1 = \beta_1 + \beta_3$ represents the combined effects of competition and disclosure that arises from increases in the number of rivals and $\gamma_2 = \beta_2 + \beta_3$ is the combined effects of indirect competition and disclosure from non-rivals that operate in related markets. $\gamma_3 = -\beta_3$ represents the effect of disclosure threats that arise from rivals and non-rivals alike. Variation in the number of rivals affected by the vulnerability identifies β_1. Likewise, identification of β_2 arises from variation in the number of non-rivals affected by the vulnerability. Identification of β_3 utilizes variation between vulnerabilities in the mode of disclosure as well as variation in the number of rivals and non-rivals affected by a vulnerability. One concern with this identification strategy is the possibility that INSTANT may be endogenous: instantly disclosed vulnerabilities may differ in some unobservable way that influences patch release times. We later relax the assumption that INSTANT is exogenous by instrumenting for it, and also explore an alternative identification strategy that does not rely upon INSTANT.

The results are presented in Table 5. Note that Eq. (4) implicitly constrains the disclosure threat from rivals and non-rivals to be similar. The χ^2 test fails to reject this constraint ($\chi^2(1) = 1.21$; p-value = 0.27). Column (1) shows that the combined effect of direct competition and disclosure from rivals, ($\beta_1 + \beta_3$), is about a 9% decrease in patch release time per rival or about 15 days. Likewise the combined effect of non-rivals, ($\beta_2 + \beta_3$), is about 10% per non-rival or about 17 days. Both these coefficients are significant at the 1% level. The effect of disclosure is about 4% (or about 7 days), which is statistically significant at the 10% level. The coefficient of β_4 suggests that a 10% increase

in quantity is associated with a 1.5% decrease in patch release times, or about 3 days. The estimates of β_1 and β_2 are about 5% (8 days) per rival and about 6% (10 days) per non-rival, respectively. These estimates are also significant at the 10% level.

Also, the estimates of the market dummies are consistent with the notion that vendors release patches earlier in more competitive markets. For instance, in column (1), the estimate of the dummy for the operating system market, with a total of 23 vendors, is −1.75 (0.70). The corresponding estimate for the application server market, with three vendors, is −0.26 (0.37).

To summarize, both rivals and non-rivals have an economically significant effect on a vendor's decision of when to release a patch for a vulnerability. Also, the effects of rivals and non-rivals are similar in magnitude; the combined effect of competition and disclosure effects both from rivals and non-rivals is about 15–17 days. Further, increases in quantity also reduce duration by about 3 days. In short, our results provide support for hypotheses 1 through 4.

6. Robustness checks

In this section, we outline some of the additional analyses we undertook to examine the robustness of our estimates to various assumptions. First we examine the robustness of our results to an alternative strategy for treating right-censored observations. Second, we check the robustness of our results to the assumption that INSTANT is exogenous by presenting the results of instrumental variable regressions. Third, since our baseline estimates require accurate measurement of RIVALS and NON-RIVALS, we show the results of another model that uses an identification strategy that does not rely on such measurement. Finally, we examine whether our results were driven by knowledge spillovers of how to patch the vulnerability rather than the effects of competition or disclosure.

6.1. Censoring

We re-estimated Eq. (4) using a Tobit model in which unpatched observations are treated as right-censored. As with our baseline random effects GLS specification, the constraint that the disclosure threat from rivals and non-rivals is identical cannot be rejected ($\chi^2(1) = 1.96$; p-value = 0.16). In column (2) of Table 5, we show the results of estimating Eq. (4) using a random effects Tobit specification. The point estimates of the effect of rivals and non-rivals (γ_1 and γ_2, respectively) are similar to that of the random effects GLS estimates and statistically significant at the 5% level. The combined effect from rivals is about 9% per rival or about 15 days while that from non-rivals is about 10% per non-rival or about 17 days. Both these coefficients are significant at the 5% level. The effect of disclosure from both rivals and non-rivals is also similar to that of the random effects GLS estimates – a statistically significant (at the 10% level) 4% or about 7 days. The coefficient of β_2, recovered using $\gamma_2 + \gamma_3$, is statistically significant at the 10% level and implies the effect of competition from one more non-rival is about 10 days. While β_1 implies that the effect of competition arising from one more rival is 8 days, it is not statistically significant. The estimate of LOGQUANTITY (β_4) suggests that a 10% increase in quantity is associated with a 1.9% decrease in patch release times or about 3 days.

We also conducted a Hausman test comparing the estimates of our baseline model in column (1) with that of the constrained Tobit model. The test rejects any systematic differences between the two specifications ($\chi^2(21) = 5.97$; p-value 0.99). Hence we conclude that assigning the maximum value of LOGDURATION to observations for which the vendor did not release a patch does not bias estimates.

6.2. Potential endogeneity of instant disclosure

As noted above, identification in Eq. (4) is based on the assumption that INSTANT is exogenous. However, it is plausible that instantly disclosed vulnerabilities may differ in some unobservable way that influences patch release times. We present results that suggest that such endogeneity, if any, does not bias our estimates. We estimated an instrumental variables (IV) specification that uses instruments for INSTANT, INSTANT * RIVALS and INSTANT * NON-RIVALS which yields quantitatively similar estimates of β_1, β_2 and β_3.[21]

We use data on the identity of the identifier of the vulnerability as instruments for INSTANT. A vulnerability can be discovered by any of the following parties: end users, vendors, CERT/CC, universities or information security consultants. Identifiers of vulnerabilities have different incentives to publicly disclose vulnerabilities. For example, a consultant may be more likely to publicly disclose vulnerabilities relative to other identifiers since such disclosure may signal the consultant's technical ability. However, end users may be more likely to work with either vendors or CERT/CC since they are primarily concerned with minimizing losses from security incidents. Hence, the sources of discovery of vulnerabilities are likely to be correlated with type of disclosure but are unlikely to be correlated with duration of patching times. We use whether the vulnerability was discovered by a consulting firm (CONSULTANT), university (UNIVERSITY) or end user (USER) to predict INSTANT. We implemented the method outlined in Wooldridge (2002). We first estimated a probit regression in which we used the sources of discovery described above and other exogenous covariates in Eq. (4) to predict instant disclosure. We used the predicted value of the probit regression as an instrument for INSTANT, and interactions of the predicted value with RIVALS and NON-RIVALS as instruments for INSTANT * RIVALS and INSTANT * NON-RIVALS, respectively, and implemented the procedure suggested in Baltagi (1995).

Column (3) of Table 5 shows the results of the random effects IV model used to estimate Eq. (4). Tests for the

[21] One alternative to instrumenting for these variables would be to use a control function approach. However, the use of control functions in the presence of binary endogenous variables requires strong assumptions to achieve consistent estimates (Wooldridge, 2007). Estimates of Eq. (4) using control functions yields qualitatively similar results to those using instrumental variables.

Table 5
Direct and Indirect Competition and Disclosure effects – Dependent Variable LOGDURATION.

Variable	Eq. (4) GLS (1)	Eq. (4) Tobit (2)	Eq. (4) IV (3)	Eq. (5) GLS (4)	Eq. (5) GLS No Unix (5)
INSTANT	−0.00	−0.02	−0.05	−0.08	0.06
	(0.37)	(0.37)	(0.92)	(0.35)	(0.44)
RIVALS ($\gamma_1 = \beta_1 + \beta_3$)	−0.09***	−0.09***	−0.08*		
	(0.03)	(0.04)	(0.04)		
NON-RIVALS ($\gamma_2 = \beta_2 + \beta_3$)	−0.10***	−0.10**	−0.09*		
	(0.03)	(0.04)	(0.05)		
INSTANT * RIVALS ($\gamma_3 = -\beta_3$)	0.04**	0.04**	0.04		
	(0.02)	(0.02)	(0.05)		
LOGQUANTITY (β_4)	−0.15***	−0.19***	−0.17**	−0.15**	−0.15*
	(0.08)	(0.08)	(0.08)	(0.08)	(0.09)
HARDWARE	−3.77***	−4.58***	−4.23***	−3.38*	−2.03
	(1.57)	(1.74)	(1.74)	(1.98)	(2.62)
HARDWARE * LOGQUANTITY	0.28***	0.34***	0.31***	0.26**	−0.01
	(0.11)	(0.12)	(0.12)	(0.13)	(0.25)
LOGVERSIONS	0.26	0.31*	0.40**	0.26	0.45*
	(0.16)	(0.18)	(0.19)	(0.18)	(0.25)
LOGSEVERITY	−0.14	−0.15	−0.17	−0.13	−0.19
	(0.14)	(0.14)	(0.13)	(0.13)	(0.17)
VENDORS ($\alpha_1 = \beta_1 + \beta_3$)				−0.10***	−0.13**
				(0.03)	(0.03)
INSTANT * VENDORS($\alpha_2 = -\beta_3$)				0.06**	0.03
				(0.03)	(0.03)
Constant	7.13***	7.85***	7.57***	7.10***	7.37***
	(1.08)	(1.15)	(1.20)	(1.14)	(1.18)
Implied estimate of β_1	−0.05*	−0.05	−0.04	−0.04	−0.10**
	(0.03)	(0.03)	(0.04)	(0.04)	(0.04)
Implied estimate of β_2	−0.06*	−0.06*	−0.05	−0.04	−0.10**
	(0.03)	(0.03)	(0.04)	(0.04)	(0.04)
N	461	461	461	461	195
R^2 (between)	0.14	–	0.14	0.15	0.18
R^2 (within)	0.08		0.07	0.07	0.12
R^2 (overall)	0.15		0.15	0.17	0.20
# vulnerabilities	241	241	241	241	158
Log likelihood	–	−902.02	–	–	–
σ_u	1.73	1.76	1.69	1.72	1.86

Notes: standard errors in parenthesis.
In columns (1) through (4) estimates include eight vendor fixed effects and three market fixed effects. In column (5) Red Hat and SCO vendor fixed effects cannot be estimated as both have now been removed from the sample.
* Significant at 90% level.
** Significant at 95% level.
*** Significant at 99% level.

power of the instruments suggest that the instruments are adequate (χ^2 statistic for INSTANT is 10.04; INSTANT * RIVALS is 64.23; INSTANT * NON-RIVALS is 64.44).[22] The coefficient estimate of RIVALS is statistically significant at the 10% level and suggests that one additional rival is associated with a 8% decline in duration times, or about 13 days. While the effect of disclosure (β_3) is no longer statistically significant, the point estimate implies that one additional non-rival or rival is associated with a decrease in vendor patch release times by 4% or about 6 days. Although the implied competition effects (β_1 and β_2) are also not statistically significant, the point estimates are similar to that of the random effects specification. The coefficient of LOGQUANTITY implies that a 10% increase in quantity is associated with a 1.7% decrease in patching time. A Hausman test comparing the coefficients of the baseline random effects model with that of the IV regression rejects any systematic differences between the estimates of the random effects and IV models (χ^2 1.72; *p*-value = 1.00). Hence the IV results suggest that endogeneity of instant disclosure, if any, does not affect our estimates of β_1, β_2 and β_3.

6.3. Measurement error of rivals and non-rivals

The model outlined in Eq. (4) relied on an accurate definition of product markets. Any measurement error in rivals or non-rivals could potentially bias the estimates of direct and indirect competition. In this section, we present the results of estimates of β_1, β_2 and β_3 obtained by solely exploiting variation in the mode of disclosure of vulnerabilities; we assume in this section that the effect of the marginal rival and non-rival are the same. This strategy does not require accurate measurement of rivals and non-rivals. However it constrains β_1 to be equal to β_2.

[22] As before, we were unable to reject the constraint imposed by Eq. (4) that the disclosure effect of rivals is the same as non-rivals ($\chi^2 = 0.39$; *p*-value 0.53).

Recognizing that VENDORS is the sum of RIVALS and NON-RIVALS, the estimating equation can be written as

$$\begin{aligned} LOGDURATION_{imv} = {} & \beta_0 + \alpha_1 VENDORS_v + \alpha_2 INSTANT_v \\ & * VENDORS_v + \beta_4 LOGQUANTITY_{im} \\ & + \beta_5 INSTANT_v + \theta_1 X_i + \theta_2 Z_v \\ & + \theta_3 K_m + \varepsilon_{iv} \end{aligned} \quad (5)$$

where $\alpha_1 = \beta_1 + \beta_3$ and $\alpha_2 = -\beta_3$. Under instant disclosure, only the direct and indirect competition effects are operative. When the vulnerability is not instantly disclosed, the disclosure effect is also operative. Hence, $\alpha_2 = -\beta_3$ identifies how increases in the number of vendors will lower patch release times through disclosure. The separate effect of competition is thus $\alpha_1 + \alpha_2$.

Column (4) of Table 5 shows the results of random effects GLS estimation for Eq. (5). The estimate of α_2 is a statistically significant (at the 10% level) 0.06, and implies that one additional vendor is associated with a 6% decline in patch release times due to disclosure threat, or about 10 days. The coefficient of α_1 implies that one additional vendor is associated with a 10%, or about 17 days, decline in patch release times due to the combined effect of competition and disclosure. The separate effect of competition (recovered using $\alpha_1 + \alpha_2$) suggests that one additional vendor leads to a 4% decline in duration, or about 7 days, due to the effects of competition. However this estimate is not statistically significant. Estimates of the effect of quantity (β_4) suggest that a 10% increase in installed base is associated with 1.5% decline in duration. Since the results of this specification are qualitatively similar to our earlier estimates, we conclude that possible mis-measurement of RIVALS and NON-RIVALS does not bias our estimates of β_1, β_2 and β_3.

6.4. Spillovers

One potential alternative interpretation of our results is that they reflect spillovers across vendors on how to fix vulnerabilities: The greater the number of vendors affected by a vulnerability, the greater the knowledge spillovers, and the lower the patch release time. Note that the spillover hypothesis does not predict that the effects of rivals and non-rivals would be greater under non-instant than instant disclosure, as we find above. In other words, the presence of spillovers cannot be used to explain our disclosure results. This makes spillovers an unlikely explanation for our results.

A sharper test that the presence of spillovers is unlikely to be the reason for our competition results can be obtained by excluding from our sample vulnerabilities that are similar to one another, for which patches are likely to be most similar. More specifically we re-estimate Eq. (5) after excluding observations that relate to UNIX and UNIX clones. UNIX is a widely used system, with different UNIX based systems having similar (though not the same) code. If spillovers were driving our results, they would be most pronounced in vulnerabilities in UNIX based products. Excluding UNIX products should therefore attenuate our estimates of the effect of direct and indirect competition.

Column (5) of Table 5 shows these results.[23] The results suggest that the direct and indirect effects are qualitatively similar to those shown in Table 5, column (4). If anything, the point estimates of direct and indirect competition are marginally higher (13% per vendor) without the UNIX related products. Hence we conclude that our results are not driven by spillovers between vendors.

7. Discussion and conclusion

We study how direct competition, indirect competition, disclosure, and market size influence decisions by software vendors to invest in the patching of software vulnerabilities, which remains key to cyber security. Our baseline results indicate that one additional non-rival has a similar impact on patching times as an additional rival – a reduction of 8–10 days. In addition, each additional rival or non-rival increases the likelihood of the vulnerability being disclosed, resulting in a reduction in patching time by about 7 days. Further, our results show that vulnerabilities for software products in larger markets are associated with faster patch release times: a 10% increase in the installed base leads to a 1.5% decline in patching times, or about 3 days. Our results fully support all of our main hypotheses.

7.1. Limitations

As with any empirical work, our conclusions are limited by our data. First, we are able only able to identify how one facet of competition influences patch release times: competition from vendors who are also affected by the same vulnerability. We are not able to separately identify the overall impact of competition on software quality. As a result, our results identify a lower bound for the effects of competition. Further, though we have shown that increases in rivals and non-rivals will reduce patching times due to the effects of disclosure, we are unable to identify whether the disclosure effect works through actual early disclosure or through the threat of early disclosure. That is, we are unable to identify whether vendors increase their investments in software patching in response to actual disclosure of a vulnerability, or increase their investments in response to expected early disclosure. However, these limitations do not influence the primary findings of this paper: that increases in the number of vendors affected by vulnerabilities influence ex-post quality provision through the effects of competition and disclosure.

Each of our models required differing identification assumptions regarding the measurement of rivals and non-rivals and their relationship with vendor patch release times. However, by estimating a variety of different models that provide very similar estimates we are able to improve the confidence in our results. Moreover, we explore the robustness of our estimates in a variety of ways, which indicates that various possible sources of bias are not significant, and alternative explanations unlikely.

[23] Observations relating to the following operating systems were removed – SUSE Linux, SCO Linux, SCO UNIX, IBM AIX, Red Hat Linux, Sun Solaris, HP-UX, Apple (BSD based operating system only) and Digital Unix.

7.2. Implications for research

Our research provides direct evidence on a question of considerable importance to academics, managers, and policymakers: the relationship between competition and software security.

We demonstrated that the threat of early disclosure has an economically significant effect on vendors patching behavior. Thus this research in part supports the results of theoretical models that argue that threat of early disclosure affects the timing of patch release for vulnerabilities.

We also advance prior empirical research on the relationship between competition and quality. We showed that quality provision is influenced not only by the number of rivals competing with the firm, but also by the number of non-rivals that were affected by the same vulnerability. This suggests that future research on the determinants of quality provision in software markets should focus in particular on the effect of changes in the number of firms that share common code bases. More broadly, we show that in software markets, vendors in one market can influence strategic behavior in another market, due to shared code. We believe that this phenomenon is more important than is widely appreciated. Recent trends in programming such as object-oriented programming and open source have emphasized software reuse. Research on software engineering has long recognized the promises and challenges of software reuse in the design and development of software. However, there has been relatively little research on how this practice influences strategic decision-making in firms. Shared code bases have the potential to influence product market strategies in unrelated markets through mechanisms other than the ones we consider: for example, other dimensions of software quality and exposure to intellectual property litigation. For example, in March 1993 SCO Group filed a well-known lawsuit against IBM for allegedly contributing proprietary SCO code to open-source Linux. This lawsuit had implications for Linux users and software developers across a variety of industries (e.g., Foley, 2003). In short, more research is needed to understand the implications of shared code base for the strategic behavior of software firms.

7.3. Implications for managers and policymakers

Understanding the relationship between competition and software quality is important for managers and policymakers. For managers of software buyers, it informs the decision when to invest in temporary countermeasures in anticipation of a patch. Understanding this relationship may also shape software purchase decisions: other things equal, changes in quality provision induced by competition may influence vendor or product choice. For producers of software, an understanding of this relationship will provide clues to competitor behavior. In particular, our results provide a better understanding of how disclosure threat influences vendor's response to vulnerabilities.

These findings also have implications for how vendors build their products. Vendors who use code shared by rivals and non-rivals should be aware of the future implications of the competition and disclosure effects for investments in ex-post quality provision. With increasing mergers among software vendors and code reuse, the vendors have to appropriately alter their software testing regime. Recent data from Symantec shows that close to a third of the vulnerabilities come from shared code (Higaki, 2008).While prior research on software engineering economics has attempted to measure how software reuse influences development costs (e.g., Banker and Kauffman, 1991; Poulin et al., 1993), our research shows that reuse will increase interdependence across firms in security countermeasures.

Last, these results have implications for the debate of how to improve software quality. Given the rapid increase in the number of reported software vulnerabilities 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. 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 optimal regulation to minimize social losses from vulnerabilities critically depends upon a proper understanding of factors that condition the timing of patch release to vulnerabilities. Our research demonstrates that despite high levels of concentration in many software markets, indirect competition and threat of disclosure from vendors in complementary markets works to reduce patching times almost as much as increases in the number of competitors.

Finally, for policy makers this work underscores the importance of disclosure threat as a valid policy instrument. In particular, our work demonstrates that disclosure threat can be effectively used to influence the timing of patch release for a vulnerability. Our results suggest that non-instant disclosure could be more welfare-enhancing than instant disclosure, and that for policy markets like CERT/CC any disclosure policy should include judicious use of disclosure threat to elicit faster vendor responses to vulnerabilities.

Acknowledgements

We thank Avi Goldfarb, Tomas Roende, and seminar participants at the University of Maryland, Carnegie Mellon University, the International Industrial Organization Conference, the ZEW, and the Workshop on the Economics of Information Security for helpful comments. We further thank CERT/CC for providing essential data. This research was partially supported by a grant from Cylab, Carnegie Mellon University, to Ashish Arora and Rahul Telang. Anand Nandkumar thanks the Software Industry Center at Carnegie Mellon University for financial support. Rahul Telang acknowledges generous support of National Science Foundation through the CAREER award CNS-0546009. Chris Forman acknowledges the support of the Sloan Foundation through an Industry Studies Fellowship. All errors are our own.

Appendix A. Supplementary data

Supplementary data associated with this article can be found, in the online version, at doi:10.1016/j.infoecopol.2009.10.002.

References

Agarwal, M., Chari, K., 2007. Software effort, quality, and cycle time: a study of CMM level 5 projects. IEEE Transactions on Software Engineering 33, 145–156.

Arbaugh, W.A., Fithen, W.L., McHugh, J., 2000. Windows of vulnerability: a case study analysis. IEEE Computer 33 (12), 52–59.

Arora, A., Caulkins, J., Telang, R., 2006a. Sell first, fix later: impact of patching on software quality. Management Science 52 (3), 465–471.

Arora, A., Nandkumar, A., Telang, R., 2006b. Impact of patches and software vulnerability information on frequency of security attacks – an empirical analysis. Information Systems Frontiers 8 (5), 350–362.

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

Arora, A., Krishnan, R., Telang, R., Yang, Y., forthcoming. An empirical analysis of software vendors' patch release behavior: impact of vulnerability disclosure. Information Systems Research.

Baltagi, B.H., 1995. Econometric Analysis of Panel Data. Wiley, New York.

Banker, R., Kauffman, R., 1991. Reuse and productivity in integrated computer-aided software engineering: an empirical study. MIS Quarterly 15 (3), 375–401.

Banker, R., Davis, G., Slaughter, S., 1998. Software development practices, software complexity, and maintenance performance. Management Science 44 (4), 433–450.

Beattie, S., Arnold, S., Cowan, C., Wagle, C., Wright, C., Shostack, A., 2002. Timing the application of security patches for optimal uptime. In: Proceedings of LISA XVI.

Borenstein, S., 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 17 (5), 611–640.

Bresnahan, T., Greenstein, S., 1999. Technological competition and the structure of the computer industry. Journal of Industrial Economics 47 (1), 1–40.

Bresnahan, T., Yin, P., 2006. Economic and Technical Drivers of Technology Choice: Browsers, Working Paper, Harvard Business School, Harvard University.

Brown, A., Booch, G., 2002. Reusing open source software and practices: the impact of open-source on commercial vendors, in software reuse: methods, techniques, and tools. In: Gacek, C. (Ed.), Seventh International Conference, ICSR-7 Proceedings. pp. 123–136.

Cavusoglu, A., Cavusoglu, H., Raghunathan, S., 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.

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

Domberger, S., Sherr, A., 1989. The impact of competition on pricing and quality of legal services. International Review of Law and Economics 9, 41–56.

Dranove, D., White, W., 1994. Recent theory and evidence on competition in hospital markets. Journal of Economics and Management Strategy 3 (1), 169–209.

Foley, J., 2003. You May Be Next, Information Week, November 23. <http://www.informationweek.com/shared/printableArticle.jhtml?articleID=16400348>.

Forman, C., Goldfarb, A., Greenstein, S., 2005. How did location affect adoption of the commercial Internet? Global village vs. urban leadership. Journal of Urban Economics 58, 389–420.

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

Hann, Il-Horn, Hui, Kai-Lung, Tom Lee, Sang-Yong, Ivan, P.L., 2007. Overcoming online information privacy concerns: an information-processing theory approach. Journal of Management Information Systems 24 (2), 13–42.

Harter, D.E., Krishnan, M.S., Slaughter, S., 2000. Effects of process maturity on quality, cycle time, and effort in software product development. Management Science 46 (4), 451–466.

Higaki, Wesley, 2008. What Are Vendors Doing To Make Software Secure? Cylab Seminar. <http://www.cylab.cmu.edu/default.aspx?id=2434>.

Honeynet Project, 2004. Know Your Enemy: Trends. <http://www.honeynet.org/papers/trends/life-linux.pdf>.

Hoxby, C., 2000. Does competition among public schools benefit students or taxpayers? American Economic Review 90 (5), 1209–1238.

Kannan, K., Telang, R., 2005. Market for software vulnerabilities? Think again. Management Science 51 (5), 726–740.

Kretschmer, T., 2005. Competing Technologies in the Database Management Systems Market, NET Institute Working Paper #05-17.

Levhari, D., Peles, Y., 1973. Market structure, quality and durability. The Bell Journal of Economics and Management Science 4 (1), 235–248.

Li, P., Rao, H.R., 2007. An examination of private intermediaries' roles in software vulnerabilities disclosure. Information Systems Frontiers 9, 531–539.

Mazzeo, M., 2003. Competition and service quality in the US airline industry. Review of Industrial Organization 22, 275–296.

Nizovtsev, D.T., Thursby, M., 2007. To disclose or not? An analysis of software user behavior. Information Economics and Policy 19 (1), 43–64.

Poulin, J.S., Caruso, J.M., Hancock, D.R., 1993. The business case for software reuse. IBM Systems Journal 32 (4), 567–594.

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.

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.

Symantec, 2004. Symantec Internet Security Threat Report 2004. <http://eval.symantec.com/mktginfo/enterprise/white_papers/ent-whitepaper_symantec_internet_security_threat_report_vi.pdf>.

Telang, R., Wattal, S., 2007. Impact of software vulnerability announcements on the market value of software vendors – an empirical investigation. IEEE Transactions on Software Engineering 33 (8), 544–557.

Tucker, C., Miller, A., 2008. Privacy Protection and Technology Diffusion: The case of Electronic Medical Records, Working Paper, MIT Sloan School of Management.

West, J., Dedrick, J., 2000. Innovation and control in standards architectures: the rise and fall of Japan's PC-98. Information Systems Research 11 (2), 197–216.

Wooldridge, J., 2002. Econometric Analysis of Cross Section and Panel Data. MIT Press.

Wooldridge, J., 2007. What's New in Econometrics? Lecture 6: Control Functions and Related Methods. <http://www.nber.org/WNE/Slides7-31-07/slides_6_controlfuncs.pdf>.