The effect of competition intensity on software security - An empirical analysis of security patch release on the web browser market[*]

Arrah-Marie Jo[†]

May 2017

Abstract

This paper examines the effect of competition intensity on software vendors' security investments. We study the impact of market concentration on software publishers' responsiveness in patching security vulnerabilities when the software is provided free of charge to users. For that, we focus on the web browser market. We first develop a formal model of competition where the user demand depends only on product quality and investigate whether equilibrium levels of quality increase as competition intensifies. Then, we test the model's predictions relying on a 10-year pooled cross-sectional data set on web browser vulnerabilities discovered and patched from 2007 to 2016. We find strong evidence that an increasing market concentration positively impacts the vendor's responsiveness in patching vulnerabilities. Nevertheless, the magnitude of this positive effect is reduced when the vendor is too dominant. In addition, we test and confirm several ideas suggested by practitioners and previous academic works related to vulnerability patching, such as the impact of vulnerability information disclosure, open source project, and software age.

1 Introduction

The discovery of highly critical vulnerabilities such as Heartbleed, Shellshock and Poodle in recent years have stressed once more the importance of vulnerability patching in addressing Internet and software security issues.[1] Despite security practitioner's efforts, it seems that attackers move faster to exploit vulnerabilities than vendors provide patches. Just considering the five most exploited zero-day vulnerabilities each year, vendors spend on average 59 days to release a patch while thousands of attacks exploiting these vulnerabilities are executed in the

[*]I thank my thesis advisor Marc Bourreau for his patient guidance and support. I also thank Rainer Böhme, Grazia Cecere, Jens Grossklags, Thomas Maillart and the anonymous reviewers of WEIS 2017 for their useful feedback and comments on a previous version of the paper.

[†]Telecom Paris Tech. E-mail: arrah-marie.jo@telecom-paristech.fr

[1]Heartbleed, Shellshock and Poodle are vulnerabilities in a cryptographic software library called OpenSSL. These vulnerabilities allow stealing information that are protected, under normal conditions, by the SSL/TLS encryption used to secure transactions on the Internet. These vulnerabilities have been extensively exploited for several years and have affected more than two third of active web sites on the Internet, partly due to the predominance of few systems. Indeed, in 2014, the combined market share of just Apache and Nginx, the two most widely spread web servers using the OpenSSL library, was over 66% (Source: http://heartbleed.com)

first month they were identified by the vendors.[2]

How can we explain such difference in the attackers' and vendors' responsiveness? Technical explanations may be part of the answer, yet in recent years, researchers have realized that economic analysis can also offer valuable insights on these issues (Anderson and Moore, 2007). An important question is to identify the economic factors affecting a firm's incentives to invest in the security of its products.[3] Do software editors have enough incentives to provide an appropriate level of security to their users? In particular, considering that markets in software industry are generally highly concentrated (Varian, 2001), does market concentration have a positive or a negative impact on software vendors' responsiveness in releasing security patches when software are provided free of charge to users? This question is all the more relevant as offering products and services free of charge to users is a major element of today digital market services (Monopolkommission, 2015). To the best of our knowledge, there is no empirical study that focuses on the relationship between competition and quality of "free" products, yet this is a dominant case in digital markets.

In this paper, we examine the relationship between the competition intensity in the market – measured by the market concentration – and the responsiveness of a software editor in releasing a security patch. We focus on a market in which the product is provided free of charge to consumers – a revenue model typical to *multi-sided market* and suffers from important security issues, namely the web browser market.

We first present a simple theoretical model where firms compete in quality and study how equilibrium security quality investments change as the number of firms competing in the market increases. The model shows that market concentration has two opposite effects on a firm's investment behavior. On the one hand, the smaller the number of firms competing in the market, the higher the equilibrium level of security investment of a firm. On the other hand, when one firm particularly dominates the market, the larger the dominant firm's market share, the lower its security investments as well as the positive effect of a reduction of competing firms. Taking account of these two aspects, we suggest that market concentration has a positive effect on the security quality provided by web browser publishers; however, when a firm dominates the market, this positive effect is less clear.

We confirm these results using a pooled cross-sectional data set of 586 web browser vulnerabilities, discovered and patched from January 2007 to December 2016. In addition, we find that software vendors tend to release security patches more rapidly when vulnerability information is disclosed to public earlier and that web browsers with open source core are patched faster than proprietary ones.[4]

Why study the relationships between software editors' investment incentives in security and competition in this particular market? We point out several reasons why it is relevant to focus on the web browser market.

First, with the increasing use of the Internet and the growth of web-related industry, it is clear that web

[2]Figures from ISTR 2015 and 2016 (Symantec's 2015 Internet Security Report p.67 and 2016's report p.40). Note that almost all of these "top vulnerabilities" affect web browsers. Moreover, according to our data (on web browser vulnerabilities identified and patched from January 2007 to December 2016), web browser publishers spent on average 103 days to release a patch for vulnerabilities assessed as "highly critical" by the National Vulnerability Database (NVD) i.e. vulnerabilities with a Common Vulnerability Scoring System (CVSS) score of more than 9/10.

[3]Another important issue related to the delay of patch installation would be the patch-update management incentives of users (Cavusoglu, Cavusoglu, and Zhang, 2008)

[4]By "core" we designate more specifically the rendering engine, which is a core component of the web browser.

browsers have a key role in Internet security. Indeed they are the principal "windows" of the World Wide Web, installed on almost all computers and devices that have access to Internet. Their wide use naturally enlarges the user's exposure to security threats; countless cyber-attack and scams have been done through web browsers or have used web browser vulnerabilities, from famous banking trojans (Zeus, Dyre...), SSL/TSL vulnerabilities (Hearbleed...), to web browser based botnets (often using Javascript). What is more, the most exploited zero day vulnerabilities each year are usually directly affecting web browsers.[5] Moreover, the core component of a web browser, namely the rendering engine, is used in almost every web application from email clients, mobile apps to eBook readers, as it an essential component in the display and edition of web content. Because of the wide use of its main component, Web security is even more dependent to the security of web browsers.

Secondly, as mentioned previously, it is a market exemplifying one of the major challenges competition policy in digital markets face today. Economically speaking, web browser publishers are multi-sided platforms where one side of the market – the users' side – is subsidized by the other – web site owners and advertisers' side –. These platforms become increasingly vertically integrated and diversified, extending their services for Internet users (search engine, email client, web feed aggregator etc.), developers (web development tools), and advertisers (advertising networks). On the other side, as users are free of charge – so their utility is not affected by the price –, they will naturally value qualitative aspects such as security. While concerns are being raised vis-à-vis the powerful market positions of some key web platforms, one can ask whether market concentration acts in the same way on firms' investment incentives in quality as it does in a market with no such situation.

In addition, several aspects strengthens the robustness of our study. First, we focus on a single market and study a relatively long period of time of ten years. Using market shares of each web browser that has existed at each quarter during the investigated period enables us to obtain accurate concentration measures based on market shares, such as the HHI and the *four-firm concentration ratio*. Moreover, in order to correctly isolate the effect of competition, we account for numerous technical and economic factors that can influence the time to develop a security patch. Our exchange with a Google security engineer helped us consider factors that software programmers themselves identify as impacting the patch release time, such as the code quality or the age of the software. We also account for factors that are often treated and mentioned in the literature of information security economics, such as information disclosure and open source projects. Furthermore, by considering only vulnerabilities that web browser publishers themselves are responsible to secure, we eliminate the possibility of not isolating the influence or other firms' action from the web browser publisher's behavior. For instance, although web browser security is significantly dependent to Adobe Flash vulnerabilities, they were excluded from our data set since patches for Adobe Flash vulnerabilities are mainly developed by Adobe and not by Web browser publishers. In other words, we only consider vulnerabilities which can be secured by web browser publishers themselves without relying on a third party publisher so that the patching time spent by the vendor depends on its own investment decision.

[5]Source: ISTR 2015 and 2016. Four of the five most exploited vulnerabilities according to Symantec annual report on Internet Security in 2014 and 2015 are web browser vulnerabilities.

The paper is organized as follows. In Section 2, we review the literature most relevant to this topic. Following that, we describe the key events that characterize the lifecycle of a software vulnerability in Section 3, then summarize the specificities of the web browser market and its revenue model in Section 4. Section 5 presents the theoretical model. We present the empirical strategy and the data in Sections 6 and 7. Estimation results follow in Section 8 and the final section provides conclusions.

2 Literature Review

This work draws principally from two streams of research: the economics of information security and competition and quality provision.

The literature on the economics of information security is recent and thriving; it aims at studying the potential market failures causing information systems insecurity. Vulnerability discovery and patch management are one of the topics at the heart of this field. Analytical works in this literature study various questions: some papers analyze the welfare implications of different liability and information disclosure policies (August and Tunca, 2011; Choi, Fershtman, and Gandal, 2010; Arora, Nandkumar, and Telang, 2006b); others investigate the coordination possibilities between software vendors and customers in the patch management process (Cavusoglu et al., 2008) or examine the role of different vulnerability discovery mechanisms (Kannan and Telang, 2005). However, few studies account for the impact of market structure on firms' information security provision. Kim, Chen, and Mukhopadhyay (2009) build a model of risk sharing for security losses between software vendors and users. They show that market competition can lead to a higher level of risk sharing, and therefore a higher level of security quality. They study the vendor's incentive to share the risks with its users with a broad definition of "risk sharing", which includes information sharing, damage cost or security investment cost sharing. In our paper, we particularly focus on the vendor's investment incentives in software security. Gal-Or and Ghose (2005) examine the effects of market characteristics such as the degree of intra-industry competitiveness and firm size on a firm's information sharing behavior and security technology investment decisions. One of their main findings is that the incentives to share information and to invest in security increase as the degree of competitiveness in an industry increases. Their model focuses on the role of information sharing, which acts as a strategic complement of firm's security technology investment and as a social welfare-enhancing mean. Arora, Caulkins, and Telang (2006a) analyze the initial security quality of a software and the patching behavior of software vendors and find that the market size has a crucial role on the vendor's behavior. Their model however considers only a market under monopoly.

A very diversified literature considers the link between competition and quality. Theoretical models often study quality as a criteria of product differentiation (Waterman, 1990; Economides, 1993; Chambers, Kouvelis, and Semple, 2006) and examine its relationship with the price (Spence, 1975; Gal-Or, 1983; Narasimhan, Ghosh, and Mendez, 1993). Some models examine the impact of the number of firms in the market on quality. In particular, Banker, Khosla, and Sinha (1998) models a demand function linear in price and in quality and show that the relation between quality and competition depends on what we define as competition (whether it consists in the number of firms in the market, difference in the intrinsic demand potential for products, difference in cost advantage between firms, or the possibility of cooperation). However, papers studying the impact of competition on quality provision when the price is not borne by consumers are very limited. Using the Salop's circular model and considering a demand function multiplicative in quality, Waterman (1990) suggests that firms invest more in quality in the case of goods free of charge for consumers. The focus of the paper is however on the effect of competition on product diversity. Argenton and Prüfer (2012) consider the effect of the user traffic on search engine competition. They assume that the larger the installed base of users, the less costly it becomes for a search engine to provide a given search quality, and show that the competitive advantage of having a larger installed base allows the dominant firm to drive out its competitors, which leads to a stable monopoly. As they do, we model competition as a simultaneous bids by firms on product quality. Empirical works approach a broad range of industries from airline industry (Mazzeo, 2003), banking (Cohen and Mazzeo, 2004), hotel industry (Mazzeo, 2002), legal services (Domberger and Sherr, 1989) and software (Arora, Forman, Nandkumar, and Telang, 2010a). All of them find that an increase in competition leads to better quality provision. However, no study deals with the case of "free" products.

So far, empirical work dealing with vulnerability discovery and patch management have essentially focused on issues related to vulnerability disclosure. Some researchers study the impact of vulnerability information disclosure to stock exchange prices (Campbell, Gordon, Loeb, and Zhou, 2003; Telang and Wattal, 2007) and other papers closer to ours on security investment (Gordon, Loeb, Lucyshyn, and Sohail, 2006; Arora, Krishnan, Telang, and Yang, 2010b). In particular, Arora et al. (2010b) exploit similar type of data as ours to investigate the vendor's responsiveness to vulnerability information disclosure and the impact of information disclosure on the frequency of cyber-attacks. Ransbotham, Mitra, and Ramsey (2008) use security alerts data from a private security service provider to examine the impact of vulnerability disclosure on the risk of cyber attacks. Our empirical model accounts for the impact of vulnerability information disclosure but it is not our main subject of interest. Information disclosure is rather considered as one of the essential factors that enable to isolate the effect of competition.

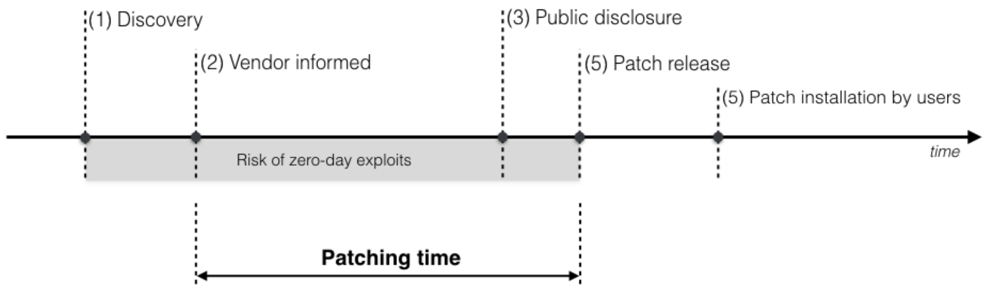
The study that comes closest to ours is of Arora et al. (2010a). Our paper has a similar purpose to them as we both analyze empirically the effect of competition on software publishers' security investment behavior. Also, both of us use the time to patch as a measure of software quality. However, our approach differs to theirs in several points. First, Arora et al. (2010a) analyze software vendors' reaction with regard to the patching behavior of other vendors that are affected by the same vulnerability. They consider that sharing a common vulnerability put vendors in a competitive situation, whether they are actually operating in the same market or not. In our case, we consider a narrower definition of competition by limiting our analysis to interactions within a single market and using market concentration as a measure of competition intensity. Secondly, as mentioned earlier, we focus on the web browser market in which vendors adopt a particular revenue model to provide the web browser free of charge to users. Lastly, we study a relatively long period of time – a ten-year-period – while Arora et al. (2010a) consider a large panel of different software during a shorter period.[6]

3 Software vulnerabilities and security quality

In information systems and computer security, a *vulnerability* refers to a weakness in the architecture, design, or code, which leaves the software or the system open to potential for exploitation.[7]

Figure 1 presents major events that may occur during the lifecycle of a software vulnerability, from its discovery (either by a malevolent or a well-intentioned actor) to its securing by the delivery of a patch and its installation. This timeline is consistent with its representation by Ioannidis, Pym, and Williams (2012), Beres, Griffin, Shiu, Heitman, Markle, and Ventura (2008), or Frei, May, Fiedler, and Plattner (2006).

Figure 1: Lifecycle of a vulnerability



Five key events outline the different phases during the lifecycle of a vulnerability: (1) the discovery of the vulnerability, (2) the vendor being informed about the existence of the vulnerability, (3) the public disclosure of the vulnerability, (4) patch release by the vendor, and (5) patch installation by the user.

The vulnerability may be discovered first by an agent that does not work for the affected software vendor. We call it then a *zero-day vulnerability*, since the software vendor has zero day left to deliver a patch before someone exploits the vulnerability. The exposure risk of the software can grow very quickly: information about the vulnerability can be kept by a discoverer who does not have any malicious intent, but it also may well be discovered by a hacker who releases an exploit and make it freely accessible on the Internet. In this context, different public and private organisms are specialized in vulnerability research and help software publishers and companies – the users of information systems – to identify the security flaws and to secure them. Some are non commercial, such as the Computer Emergency Response Teams (CERT), others have a commercial purpose, operating in the *market for vulnerabilities*. In the commercial side, security firms also called *security infomediaries* pay the researchers for the discovery of vulnerabilities and exchange with vendors to provide

[6]Our data covers a 10-year-period from 2007 to 2016 while Arora et al. (2010a) consider a 4-year-period from 2000 to 2003.
[7]Source: `https://cwe.mitre.org`

protection solutions for their clients.[8] Furthermore, an increasing number of software vendors manage their own security research team or offer bounties for vulnerability discovery. It is precisely these vulnerability research programs which are the main data sources of our study, namely Tipping Point's Zero Day Initiative (ZDI) program, Verisign's iDefense Vulnerability Contributor Program (VCP), and Google Project Zero. ZDI is a program run by the security firm Tipping Point since 2005 and iDefense VCP is the corresponding program of Verisign initiated in 2003. When ZDI or iDefense VCP identify a vulnerability, they inform the affected vendors about its existence and communicate with them until they release a patch. Google Project Zero is a more recent project initiated by Google in 2014 that also financially rewards researchers for vulnerability discovery. Google Project Zero focuses especially on vulnerabilities that affect Google's own product and its competitors.

Vulnerability information can be publicly disclosed before the vendor releases a security fix. Though the public disclosure of vulnerability information increases the exposure to attack, many advocate that disclosure makes the vendors more responsive in patching their software (e.g., see Arora et al. (2006a)). Vulnerability research organisms generally apply a policy of *responsible disclosure*, notifying the affected vendor and keeping the vulnerability information confidential during a period of time. For instance, ZDI keeps the vulnerability information confidential for 4 months, iDefense for 6 months and Google Project Zero for 3 months. Our study accounts for the impact of vulnerability information disclosure although it is not our main focus. It is rather used as one of the controls that help to identify the effect of the competitive pressure.

Lastly, a system affected by the vulnerability will be exposed to security risks until the user actually installs the patch. Software are often left unpatched, sometimes for months, whether it is by an individual user or a firm. For firms, it is mainly for operational reasons (Cavusoglu et al., 2008; Shostack, 2003), such as the concern that a patched version of the software may conflict with other existing applications or because patch installation requires to interrupt the production environment while business continuity is crucial.

From the moment the vulnerability is discovered, each day that passes without a patch increases the exposure risk of the software (Schneier, 2000; McGraw and CTO, 2004). In other words, the security quality of a software depends considerably on how quickly the vendor releases a security fix (Arora et al., 2006a). In line with this idea, we consider the promptness of the software vendor to provide security patches as a proxy of the software's security quality. More precisely, we measure the duration between the moment the software publisher is informed about the existence of the vulnerability and when it releases a patch. This duration corresponds to $(4) - (2)$ of the timeline in Figure 1.

Two main considerations will affect the vendor's patching decision: the cost to develop the patch and the extent to which he internalizes the user losses related to the security flaw (Arora et al., 2010a). In more practical terms, the vendor will consider a number of factors such as the importance of the flaw in terms of security, the impact on its reputation especially when the information is disclosed to public, the complexity to fix the

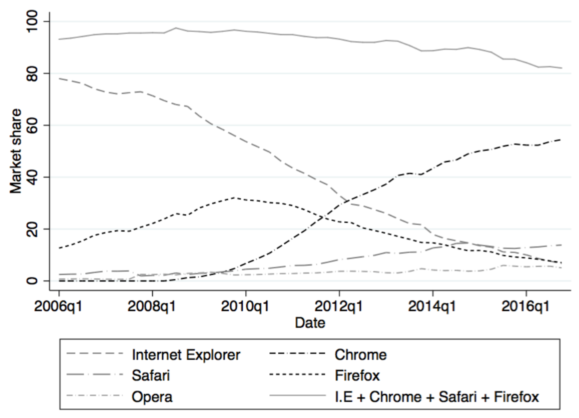
[8]iDefense and Tipping Point are the two major security firms playing in this market.

flaw, the human resources available to develop the patch and the competitive pressure. In our paper, we are particularly interested in the impact the presence of competitors may have on the vendor's patching decision, when other things are put equal.

4 The Web browser and its revenue model

A web browser is a software that gives access to information resources on the World Wide Web (WWW). Its primary role is to retrieve and display content from remote web servers using the HyperText Transfer Protocol (HTTP). Today, the most popular web browsers are Google Chrome, Mozilla Firefox, Microsoft Internet Explorer, and Apple Safari. Figure 2 shows the evolution of their market shares from 2006 to 2016.

Figure 2: Web browser's market share from 2006 to 2016



From an economic point of view, a web browser is a multi-sided platform where web surfers and webpage owners meet. Several aspects of this type of market are of particular interest to us. First, it is the use of a free pricing strategy. Free pricing strategy is very frequent in software markets where complementarity of products and cross-side network externalities are significant (Rochet and Tirole, 2006; Evans, 2003). Table 1, which compares some characteristics of the four most popular web browsers, suggests that either a publisher is vertically integrated and has its own search engine, or its revenue comes mostly from a partnership with a search engine. Indeed nowadays, any words or hyperlinks we type on a web browser's search bar results in a request on a search engine. In other words, web browser publishers derive their revenue from search engines that in turn make use of the browsing traffic and get payed through their advertisement platforms (E.g. Google Adsense, see Figure 3).

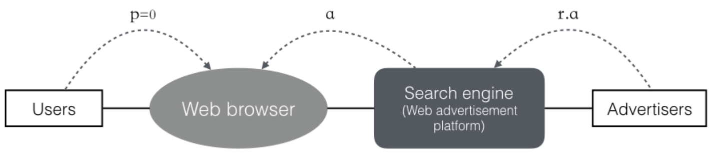
Secondly, the web browser market is concentrated due to network effects and economies of scale. Indeed, more than 150 web browsers were created since the first one in 1990, but a large part of the market is continuously shared by a small number of publishers.[9] One may think that market concentration is harmful to software editors' security investment incentives. This concern becomes all the more relevant as web browser editors increasingly employ strategies aiming at user lock-in, which is likely to confer a potential ex-post power to

[9]From 2005 to 2016, more than 75% of the market is shared by the four most popular browsers. Source: Wikipedia

Table 1: Comparison of the four most popular web browsers

Browser	Publisher	Rendering engine	License	Revenue model
Chrome	Google	Blink (fork of Webkit)	Proprietary software with open source rendering engine (GNU LPGL). An open source version of the browser is available (Chromium)	90% of ABC(Google)'s revenue come from search related ad.
Firefox	Mozilla	Gecko	Open source (MPL)	Built-in search engine royalties ($>$ 90% of whole revenues, $\simeq$100M$) and donations. Exclusive contract with Google until 2014 and with Yahoo Search since 2015
Internet Explorer	Microsoft	Trident and EdgeHTML since 2015	Proprietary	Revenues from other activities
Safari	Apple	Webkit	Proprietary software with open source rendering engine (GNU LPGL)	1B$ of built-in search engine royalties from Google (in 2014)

Figure 3: Web browser's revenue model



Users do not pay for the web browser thus $p = 0$. The editor derives its revenue from the search engine (SE), which in turn gets a revenue from the advertisers that pay for user traffic. Denoting by a the revenue the web browser gets per user, the SE gets $r \cdot a$ from the advertisers where $r > 1$.

them (Monopolkommission, 2015). On the contrary, it could be a natural behavior for firms to compete in quality in order to preserve the market (Bork and Sidak, 2012). Especially, here the consumer does not bear a financial cost when using a web browser so his utility is not a function of the price but of other factors that he values. For instance, when a user has to choose a web browser among others, he will evaluate how much it is secured (protection from data breaches, viruses, malware..), how fast it is (browsing and rendering speed), and how user-friendly it is. Considering these , Google has heavily communicated on its web browser's simplicity, performance and security since its launch.[10] Despite its increasing dominance (see figure 2), Google has also contributed to improving web browser's quality, especially concerning the rendering speed of videos and dynamic content (Anand and Saxena, 2013; Smedberg, 2010). In light of this, the free pricing facet seems important to take into account when examining the relationship between competition and web browser publishers' security provision behavior.

Lastly, what we designate as quality necessarily comprehends some subjective criteria. The appreciation of a software quality may depend on a personal taste, the type of use, or the level of technical knowledge the user has. For instance, depending on whether it is a profane or sophisticated user, the user will value the simplicity of use or the modularity. However, we note that the insecurity of a software will indubitably have a negative impact on the value of the software, no matter the type of user.

[10] "Speed, simplicity and security are the key aspects of Google Chrome": When launching Google Chrome, the official press events and press release communicated about Chrome web browser using key words such as "lightweight", "fastest", "speed", "simplicity", "secure", "open source". Source: https://googleblog.blogspot.fr/2009/11/releasing-chromium-os-open-source.html

5 A model

We have seen that, like in many other software markets, web browser publishers offer their software free of charge to users and derive revenues from a neighboring market that make use of the product usage.

The question we would like to answer in this section is whether the security quality of a free software increases or decreases as competition intensifies. For that purpose, we develop a formal model of oligopolistic competition where firms compete in quality and study how equilibrium levels of quality change as the number of firms competing in the market increases. The web browser market motivates our study; however, the model would also fit to other markets with a similar revenue model.

We assume that consumer's utility depends only on the security quality and that the revenue per user that the software publisher derives is exogenous. Moreover, we consider the possibility for a firm to have an installed base of loyal consumers.

We consider n firms, labeled 1 to n. Each firm $i \in \{1, \dots, n\}$ offers one product - a web browser -, for which it must choose a security quality level s_i. We assume that the security quality s_i is measurable, with values in $[0, \infty)$.

On the demand side, there is a fixed unit mass of consumers. We consider indeed that the market is fully covered and the aggregate demand on the market is fixed. This is consistent with the case of a mature market such as the web browser one, where every Internet user has already acquired the product (so there is no new demand) and there is no cost that stands in the way of using the product (so aggregated demand does not decrease).[11]

Similar to Argenton and Prüfer (2012), we model competition as a tournament between web browsers with simultaneous security quality choices. More precisely, demands are allocated to the firms in proportion to their relative security quality.

Moreover, each firm has an installed base of loyal consumers $b_i \in [0, 1]$, where $\sum\limits_{i=1}^{n} b_i \leq 1$. Loyal consumers stick to their choice of web browser whereas non-loyal consumers can switch. Indeed, consumer loyalty (or inertia) can be due to a trust gained from a continuously good quality in former periods – This effect could be intensified if consumers are imperfectly informed about the current security quality – as well as due to the existence of switching costs (Klemperer, 1987). Although no contractual or compatibility cost exist in the case of web browsers, changing from a web browser to another may incur important migrating costs and the loss of connected services.

We assume that the marginal cost is equal to zero, which is a standard assumption in the literature on information goods (e.g, Arora et al. (2006a); Kim et al. (2009); Choi et al. (2010)). Moreover, as it is standard in studies on quality provision (See for instance Allen (1984); Ronnen (1991)), we consider an increasing and convex cost function for security quality investments. More precisely, a firm i has to invest $\phi s_i^2/2$ to deliver a the security quality level s_i, where ϕ is the fixed cost parameter for quality investments.

[11]Rochet and Tirole (2006) defines it as a market with "pure usage externalities."

Denoting by a the per-capita revenue and $B = \sum_{j=1}^{n} b_j$ the total installed base of loyal consumers in the market, firm i's profit is then:

$$\pi_i(q) = a \left[b_i + (1-B) \frac{s_i}{\sum_{j=1}^{n} s_j} \right] - \frac{\phi s_i^2}{2} \tag{1}$$

The first-order condition (FOC) with respect to s_i is:

$$\frac{\partial \pi_i}{\partial s_i} = a(1-B) \frac{\sum_{j=1}^{n} s_j - s_i}{(\sum_{j=1}^{n} s_j)^2} - \phi s_i \tag{2}$$

Since $\partial^2 \pi_i / \partial^2 s_i = -2a \sum_{\substack{j=1 \\ j \neq i}}^{n} s_j / (\sum_{j=1}^{n} s_j)^3 - \phi < 0$, the second-order condition is always satisfied. Solving for the FOC, we obtain the symmetric equilibrium security quality level for each firm:

$$s_i^* = \sqrt{(1-B) \cdot \frac{n-1}{n^2} \cdot \frac{a}{\phi}} \tag{3}$$

The following proposition outlines the main insight of this simple model:

Proposition 1. *For $n \geq 2$, the security level provided by a vendor increases with the number of competitors.*

First, equation 3 shows a non linear relationship between the number of competing firms and the equilibrium security level. The security level provided by a monopolist is zero and for $n \geq 2$, vendors compete in quality to attract unattached consumers. As the number of firms in the market decreases, there is a larger share of unattached consumers each firms can get. Thus firms invest more in security to get the share of consumers made available by the absence of an additional competitor. All in all, the larger the number of firms n, the lower the equilibrium level of security investment of each firm, yet it is always greater than the monopoly case.

If we assume the existence of loyal consumers, then the security level provided by a vendor depends not only on the market concentration but also on the total share of loyal consumers B. This corresponds to the intuitive idea that the existence of consumers less responsive to quality changes reduce the firms' incentives to provide a better security. More specifically, let's apply equation 3 to the case of a firm that highly dominates the market. We assume that firm i's share of loyal consumer is large enough to ignore the effect of the rest of the market, that is, $B = b_i = \alpha_i m_i$, where $m_i \in (0, 1]$ the firm's total market share and $\alpha_i \in (0, 1]$ the share of loyal consumers among firm i consumers. Then:

Proposition 2. *The security quality chosen by a highly dominant firm i decreases with respect to its market share ($\frac{\partial s_i}{\partial m_i} < 0$), and is always lower than the security level it would provide in a situation without an installed base.*

Proposition 3. *The larger the market share of the highly dominant firm, the lower the positive effect of market concentration on the security level it provides* ($\frac{\partial s_i}{\partial m_i \partial n} < 0$).

6 Empirical specification

Our goal is to examine the effect of competition intensity on the time a software publisher spends to release a patch, taking account other exogenous factors that can have an impact on its responsiveness. Relying on the theoretical results obtained above, we particularly focus on two aspects of competition: market concentration and the dominance of one firm.

First, to estimate the effect of market concentration on the responsiveness of the vendor affected by the vulnerability, we define the following equation:

$$\begin{aligned} Patching\_time_{ijt} = \beta_0 + \beta_1 Concentration_t \\ + \beta_2 Vuln\_char_i + \beta_3 V\&Soft\_char_j + \beta_4 Disclosure + \beta_5 Time\_trend_t + \epsilon_{ijt} \end{aligned} \tag{1}$$

where $Patching\_time_{ijt}$ is the time spent by the web browser publisher j in releasing a security patch for a given vulnerability i discovered at time t. $Concentration_t$ is a measure of the market concentration at time t. We test two different measures of concentration: the additive inverse of the number of firms competing in the market (-n) and the Herfindahl–Hirschman Index (*HHI*). In line with Proposition 1 that suggests a positive effect of market concentration on the security level provided by a vendor, we expect a negative sign for the parameter β_1. Additionally, we control for a list of other factors that may have an impact on the responsiveness of the web browser publisher in releasing a security patch.

First, $Vuln\_char_i$ is a vector of variables accounting for the characteristics of the vulnerability such as its severity or its type. On the one hand, developers may be more or less rapid to find a secure solution according to the type of the vulnerability. On the other hand, a vendor would conceivably patch more rapidly a vulnerability that has a severe security impact on the product. These variables reflect the security risks the vulnerability implies and categorize the complexity to develop the patch. Specifically, we use two types of information: the Common Vulnerability Scoring System (CVSS) and the Common Weakness Enumeration (CWE). The CVSS is a value ranging from 1 to 10 that ranks vulnerabilities according to the degree of threats they represent. The CWE categorizes software weaknesses into general classes in order to facilitate the discussion and the coordination between security practitioners when dealing with software vulnerabilities.[12] CVSS values are directly used as the *vulnerability_severity* variable, while the CWE catalog is used as a vector of dummies.

Secondly, we control for the characteristics of the web browser and its publisher by $V\&Soft\_char_j$. This

[12]In the case of web browser vulnerabilities, referenced CWEs concern the improper restriction of operations within the bounds of a memory buffer, the improper control of generation of code, configuration issues, information exposure, improper input validation, numeric errors, security features, improper limitation of a pathname to a restricted directory, permissions, privileges, and access controls, concurrent execution using shared resource with Improper synchronization, ressource management errors, and the incorrect type conversion or cast. Source: MITRE

vector of variables consists in the $software\_age$ variable, the *open_source* variable, and dummy variables assigned to each web browser publisher. In the regressions, we do not include the vendor and the *open_source* dummies together because *open_source* is highly correlated with the vendor dummies. We use them separately as different estimations to check to robustness of our model.

The $software\_age$ variable accounts for the difficulties to fix a flaw due to the code quality of the affected version. Indeed, according to practitioners, it is generally extremely complex to identify the root cause of a vulnerability, the principal reason being the quality of the code, which generally depends on how old are the software and the development tools the original programmers have used.[13]

The *open_source* variable is a dummy variable which is equal to 1 if at least the rendering engine of the affected web browser has an open source license. A stream of work argues that open source providers offer better quality and are more responsive to customer needs (Arora et al., 2010b; Lerner and Tirole, 2002). We thus expect *open_source* to reduce the *patching_time*.

We also control for other vendor-specific characteristics by a vector of dummies. Indeed, vendors may have specific patch release policies that affect their patch release decisions but are unobserved by us. Some of them may care more about their reputation because of the spillover effect on their other products. The patching time can also depend on their financial ability to invest in security (for example, in order to afford a larger security development team). In Arora et al. (2006a), this gap between vendors is controlled by the vendor size. In our case, web browser publishers present heterogeneous forms of organization and business models, from open source foundations to multinationals listed on stock exchange. The size of a firm is hence not an adequate information to account for the vendor specific unobserved factors and we preferred to attribute a dummy variable for each vendor.

Third, $Disclosure$ captures the impact of vulnerability information disclosure on the responsiveness of vendors. It is a dummy variable taking the value 1 if the vulnerability is disclosed to public from 0 to 89 days after being reported to the vendor; that is, earlier than the conventional 90 days stated by the disclosure policy of each vulnerability search programs from where our data set comes. We attribute the value 0 if the vulnerability is disclosed to public after 90 days. Indeed, a recent stream of literature in information security economics deals with the impact of information disclosure in security provision (Nizovtsev and Thursby, 2007; Arora, Telang, and Xu, 2008; Arora et al., 2010b), supporting theoretically and empirically the idea that information disclosure helps vendors to be more responsive.

Lastly, $Time\_trend$ captures a potential time trend, such as the growing awareness both among consumers and practitioners about security issues, which in turn, makes the editors more responsive in patching vulnerabilities. As a proxy of the time trend, we use the variable $qyear$ which is the quarter in which the vulnerability i is discovered. ϵ_{ijt} represents the unobservable error term.

Then, to estimate whether the effect of market concentration is identical when the affected firm highly

[13]Source: interview with Nicolas Ruff, security engineer at Google Security.

dominates the market, we specify a variant model as follow:

$$\begin{aligned} Patching\_time_{ijt} = \beta_0 + \beta_{1a}Concentration_t + \beta_{1b}Big\_mshare_{jt} + \beta_{1c}Concentration_t \cdot Big\_mshare_{jt} \\ + \beta_2 Vuln\_char_i + \beta_3 V\&Soft\_char_j + \beta_4 Disclosure + \beta_5 Time\_trend_t + \epsilon_{ijt} \end{aligned} \tag{2}$$

In this equation, we added on the baseline model (equation 1) the *Big_mshare* variable and an interaction term of this variable with *Concentration*. This variable captures the effect for a firm to have a relatively large installed base compared to its competitors at the time it is informed about the security failure. More precisely, *Big_mshare* is a dummy variable equal to 1 if the affected web browser publisher's market share is greater than M at the moment the vulnerability is discovered, where $M \in \{0.40, 0.45, 0.50\}$.[14] The coefficient of the interaction term β_{1c} represents the difference in the effect of market concentration between the case the affected firm highly dominates the market or does not. We expect β_{1b} to be positive according to Proposition 2 , and β_{1c} to be positive according to Proposition 3.

Our models are estimated with ordinary least squares (OLS) and negative binomial regressions.

7 Data and method

For the purpose of our empirical analysis, we have constructed a 10-year pooled cross-sectional data set consisting of web browser vulnerabilities identified and patched from January 2007 to December 2016. We combine data from a variety of sources: (1) vulnerability information collected from vulnerability research programs, (2) patch and update release dates of each web browser from publishers' websites, (3) quarterly market shares of each web browser from Statcounter.com, (4) additional vulnerability information from the National Vulnerability Database (NVD), and (5) the number of Internet users from the ITU ICT indicators database.

Our data come principally from private vulnerability research programs, namely Tipping Point's Zero Day Initiative (ZDI), iDefense's Vulnerability Contributor Program (iDefense), and Google Project Zero. ZDI is a program run by the security firm Tipping Point since 2005. iDefense is the corresponding program of Verisign initiated in 2003. As mentioned in Section 3, ZDI and iDefense are the two main players in the commercial vulnerability market, which financially reward security researchers in exchange for information they have about security flaws that were unknown before. Both ZDI and iDefense notify the affected vendors and communicate with them until they release a patch. Information about the vulnerability is kept confidential during a period of time specified by their disclosure policy. The disclosure timing is not strictly applied, but the actual duration of confidentiality is revealed on the program's website after the security patch is released. Similar to these programs, Google Project Zero is a project initiated by Google in 2014, which focuses on vulnerabilities that affect directly or indirectly Google's own products, meaning that a flaw affecting Internet Explorer can be treated by the project as much as a vulnerability in Google Chrome. The program also discloses the vulnerability information

[14]We limit estimations on these three values because of correlation problem with *Concentration* variable.

to public once a patch has been released or after a 90-day-deadline.

From these three programs, we collected the information related to the web browser vulnerabilities they identified. We consider only vulnerabilities specifically assigned to web browser publishers, i.e. that can be secured by the web browser publishers themselves without a third-party intervention. For instance, Adobe Flash vulnerabilities are not taken into account. Though web browsers are impacted by these vulnerabilities, the patches are mainly developed by Adobe and not by the web browser publishers.

In our analysis, one observation consists in a pair of vulnerability and the associated web browser publisher. For vulnerabilities that affect more than one web browser, we have duplicated the vulnerability information in several observations, each associated to a distinct publisher, in order to assess each publisher's strategy separately.

The time a software vendor spends in patching a vulnerability – the *patching_time*, which is our dependent variable – corresponds to the duration between the date at which the vulnerability is reported to the vendor and the the delivery date of the associated patch. The patch release date is collected from release notes of the web browser's official website. The notification date of the vulnerability to the publisher is collected from the vulnerability research programs. These programs are actually some of the few sources that provide publicly such information, given its highly confidential nature especially from the software vendor's point of view.

For each vulnerability-browser pair, we collect from Statcounter.com the market share of the affected web browser at the date the vulnerability was reported to the publisher. We also determine the number of firms in the market and the *Herfindahl–Hirschman Index* (HHI) at each quarter from January 2007 to December 2016 using these data.[15] Market shares and market concentration measures are used as our main explanatory variables.[16][17]

This database has been completed with information about vulnerability characteristics, such as the severity (CVSS) and the type of the vulnerability (CWE) from the National Vulnerability Database (NVD).[18] As an important part of our data - such as the characteristics of a vulnerability - is collected from NVD, we only retained NVD-listed vulnerabilities in our regressions.

The public disclosure date of vulnerability information is determined through a variety of sources, including the date indicated by the vulnerability research program and the release date on NVD. From each web browser's website, we collected information about each version in order to determine how old the affected version of the web browser is ($software\_age$ variable). Lastly, we have collected the number of Internet users during the tested period from ITU ICT Indicators database.

[15]In order to determine the number of firms in the market, we count a browser publisher as "present" in the market if it has more than 0.5% of the market at the time the vulnerability is discovered. Thus the number of firms is the number of browsers that have more than 0.5% of market share at the date the vulnerability is discovered.

[16]Though our data allow to compute the four-firm concentration ratio C_4 (sum of the four largest firms' market share) at each quarter, we do not use it as an explanatory variable because it is too correlated with the time.

[17]Market shares are not used as is but as a dummy which identifies whether a vulnerability is associated to a "highly dominant" web browser publisher.

[18]The Common Vulnerability Scoring System (CVSS) is a scoring system that aims at prioritizing the vulnerabilities according to threats they represent. Scores are calculated based on a formula that depends on several metrics that approximate ease of exploit and the impact of exploit. They range from 0 to 10, with 10 being the most severe. The Common Weakness Enumeration (CWE) is a categorization of software weaknesses. The dictionary is maintained by the MITRE Corporation.

Descriptive statistics

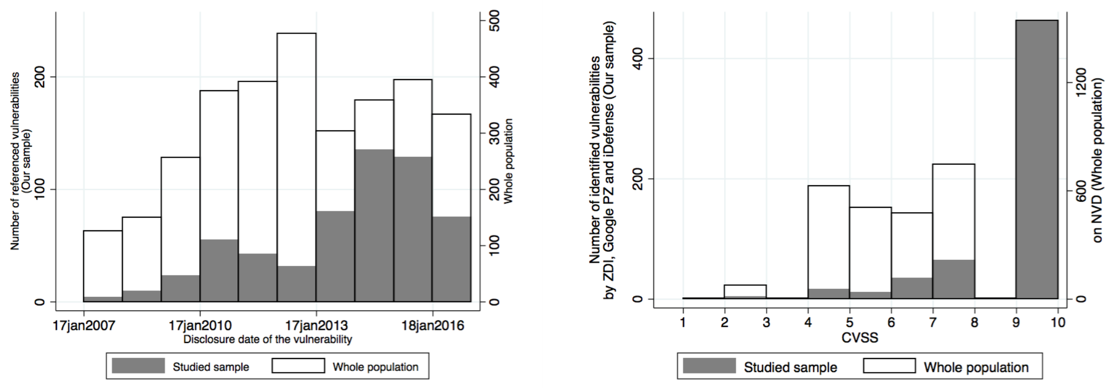
Our analysis is based on 586 observations consisting in web browser vulnerabilities identified from January 2007 to December 2016 by the three vulnerability research programs we mentioned earlier. Table 2 reports basic statistics comparing our data set to the population of web browser vulnerabilities referenced on NVD.

Our data set represents 15.5% of the total web browser vulnerabilities referenced on NVD during the investigated period, in a balanced manner over time (Figure 4 left). Nonetheless we note that high severity (CVSS of 9 to 10) vulnerabilities are significantly represented in our data compared to the parent population. We explain this bias by the commercial nature of the programs from where our data come (Figure 4 right). This bias is controlled in our regressions by the *vulnerability_severity* variable.

Table 2: Comparative statistics between the studied data set and the NVD-listed vulnerabilities

Variable	Web browser vulnerabilities listed in NVD	Our data
Number of vulnerabilities affecting...		
Apple Safari	550 (14.5%)	63(10.8%)
Google Chrome	1148 (30.3%)	32 (5.5%)
Microsoft Internet Explorer	1086 (28.7%)	441 (75.3%)
Mozilla Firefox	898 (23.7%)	60 (10.2%)
Other web browsers	81 (2.1%)	0 (0%)
Total number of vulnerabilities	3783	586
Mean of *vulnerability_severity*(CVSS)	7.25	8.75

Figure 4: Comparison of the studied data set to the whole population of NVD-listed vulnerabilities



The description of all the variables and summary statistics are reported in Table 6 and Table 7 in Appendix.

The dependent variable *patching_time* is a positive integer as the time spent by the vendor to release a patch is measured in number of days. Although it is a count variable, we note that it takes a wide range of values from 0 to 302 and the mean and the median are distant from 0.[19] Thus we consider that both OLS and count models are suitable to estimate our specifications. Poisson regression would be a baseline model for count data. However, given the presence of significant over-dispersion in our data set, we use standard deviation superior to the mean negative binomial to estimate our models.

[19]When the mean of the outcome variable is relatively high (often defined as greater than 10 as a rule of thumb), OLS regression can typically be applied to a count outcome with minimal difficulty.

Another option would have been to use a survival model. Although it is clearly possible to use a proportional hazard model here, we do not employ it because our data set is only composed of vulnerabilities which has been patched. For this reason, we do not obtain any additional information from using a survival model.

Figure 5: Evolution of market concentration in the web browser market

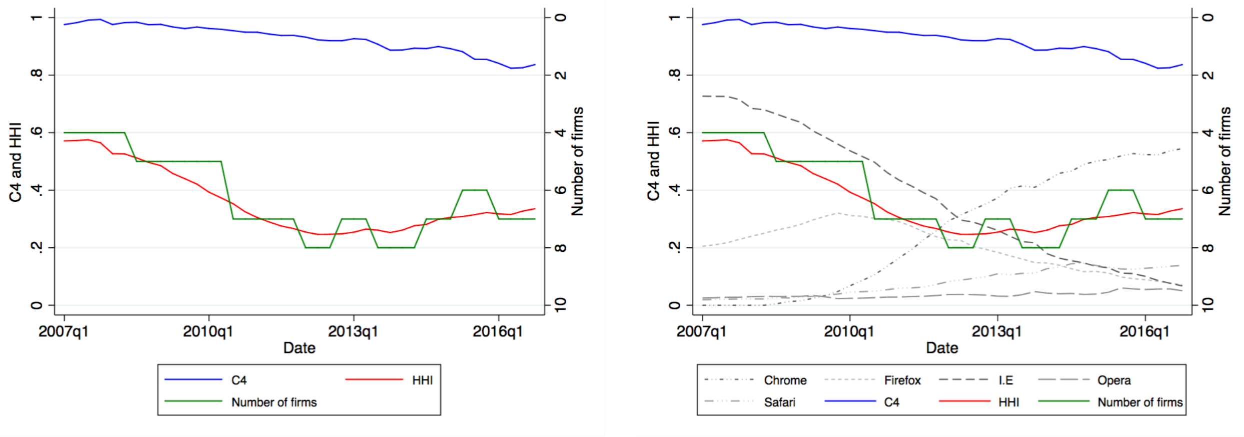


Table 3: Correlation matrix of market concentration measures

Measures	$-n$	C_1	C_4	HHI	$qyear$
$-n$	1.00				
C_1	0.75	1.00			
C_4	0.39	0.01	1.00		
HHI	0.86	0.88	0.43	1.00	
$qyear$	0.45	0.15	0.95	0.57	1.00

Figure 5 shows the evolution of different measures of market concentration in the web browser market. Although it is possible to compute the one-firm concentration ratio C_1 or the four-firm concentration ratio C_4 with our data, we only use the number of firms and the HHI as the $Concentration_t$ variable. Indeed, regulators typically rely on the HHI to measure market concentration. C_4 is also widely used in the economic literature, but rather because the HHI requires the complete distribution of the market. We also note that some dynamic aspects of the market such as the change in the dominant firm or the existence of web browsers with small market share are not well reflected in the C_4 compared to the number of firms or the HHI.[20] Moreover, it is highly correlated to the $Time\_trend$ variable (see Table 3).

8 Estimation results

To begin with, we report in Table 4 the regression results for model (1) which estimates the effect of market concentration on the patch release time. In each pair of columns, we report both OLS and NB regression results. We use two different measures of concentration, i.e. the number of firms and the HHI. Consistent with our prediction, the coefficients (for OLS) and the average marginal effects (for NB) of $Concentration$ are negative

[20]More than hundred web browsers have been introduced in the market since the creation of the first web browser in 1990

and statistically significant for each regressions. For instance, web browser publishers are likely to release a security patch about 5 days earlier when one less firm competes in the market.

Table 4: Results for model (1) using -*n* and *HHI* as concentration measures

Using as main explanatory variable (*Concentration*):	-*n*		*HHI*	
	OLS	NB	OLS	NB
Concentration	-5.483**	-4.794**	-85.35**	-114.3***
	(2.422)	(2.314)	(42.14)	(42.00)
Vulnerability specific effects				
vulnerability_severity	-5.210**	-5.308**	-5.704***	-6.219**
	(2.050)	(2.567)	(2.188)	(2.593)
vulnerability_type dummies				
*cwe*119 (Improper Restriction of Operations [...])	-6.874	-3.904	-7.152	-3.399
	(10.65)	(10.50)	(10.63)	(10.46)
*cwe*17 (Source code)	-26.67	-34.90	-26.70	-33.33
	(29.80)	(25.06)	(29.60)	(25.60)
*cwe*94 (Improper control of generation of code)	20.46	22.53	18.83	21.32
	(14.84)	(14.97)	(14.68)	(14.78)
*cwe*200 (Information exposure)	-21.65	-16.96	-24.92*	-19.24
	(14.22)	(17.10)	(14.57)	(16.57)
*cwe*20 (Improper input validation)	-7.870	0.0355	-8.294	-0.566
	(12.54)	(13.71)	(12.52)	(13.60)
*cwe*189 (Numeric errors)	-22.46	-21.90	-22.23	-19.56
	(15.64)	(14.52)	(15.46)	(14.91)
*cwe*254 (Security features)	-10.81	-7.676	-10.64	-6.901
	(13.67)	(47.85)	(13.64)	(48.13)
*cwe*22 (Improper limitation of a pathname [...])	-68.12***	-98.82***	-66.44***	-98.80***
	(14.24)	(2.854)	(14.08)	(2.874)
*cwe*264 (Permissions, privileges, and access controls)	-18.10	-18.25	-19.77	-19.29
	(14.02)	(13.22)	(14.17)	(12.99)
*cwe*362 (Concurrent execution [...])	-28.56	-39.82*	-31.88*	-41.73*
	(20.29)	(23.71)	(18.84)	(22.86)
*cwe*399 (Ressource management errors)	8.639	15.90	7.956	16.68
	(11.79)	(11.77)	(11.74)	(11.79)
*cwe*704 (Incorrect type conversion or cast)	-8.606	-8.168	-7.544	-5.299
	(11.15)	(46.92)	(11.13)	(48.28)
Soft. and vendor specific effects				
software_age	0.755*	0.795	0.789*	0.798
	(0.438)	(0.530)	(0.442)	(0.527)
apple	-10.11	-13.92**	-11.54*	-14.95**
	(6.888)	(6.838)	(6.884)	(6.696)
google	-23.21*	-31.13***	-23.46*	-32.61***
	(12.16)	(7.648)	(12.01)	(7.526)
mozilla	-23.98***	-26.37***	-24.65***	-27.78***
	(8.303)	(6.776)	(8.172)	(6.645)
Disclosure effect				
earlier_disclosure	-48.64***	-48.37***	-48.99***	-49.04***
	(3.611)	(4.462)	(3.631)	(4.474)
Time effect				
qyear	-1.513***	-1.513***	-1.605***	-2.276***
	(0.293)	(0.293)	(0.316)	(0.659)
Constant	458.7***		549.1***	
	(71.85)		(90.34)	
Observations	586	586	586	586
R-squared	0.388		0.386	
Wald chi-squared		236.43		239.51
VIF for *Concentration*	1.40		1.68	

Robust standard errors in parentheses
*** p<0.01, ** p<0.05, * p<0.1
For OLS regressions, coefficients are reported. For NB regressions, average marginal effects are reported.

Regarding other variables, the estimated impact of vulnerability severity score (*vulnerability_severity*) is negative and statistically significant, meaning that editors are likely to respond faster to more severe vulnerabilities. Besides, by regressing the *vulnerability_severity* by *vulnerability_type* dummies, we note that there is a strong relationship between the type of the vulnerability and the severity score (see Table 11 in Appendix). The characteristics of the vulnerability (its severity, its type) may influence the patching time not only because it has an impact on the intent of the vendor (prioritization according to the severity, impact on its reputation) but also because of technical reasons (difficulties to develop the patch for some types of vulnerability, learning effects on types that are frequently identified...). Our model does not study further the relationship between vulnerability characteristics and the patch release time as we focus specifically on the impact of the competition intensity. As to variables accounting for software characteristics, $software\_age$ has a positive coefficient, meaning that patches are released slower when the software version is older. We also note that Mozilla Firefox vulnerabilities are patched in average 25 days faster than other web browser ones. Estimation results using the *open_source* variable are reported in Appendix as a robustness check (Table 9). We find that *open_source* is highly significant and negative, confirming the idea suggested by the literature about the benefits of open source projects. That is, for a web browser, having at least an open source rendering engine reduces the patching time by more than 15 days. Lastly, the *earlier_disclosure* variable's coefficient is statistically significant and negative, meaning that all else the same, disclosing the vulnerability information to public shortens the patch release time. More precisely, earlier public disclosure is likely to shorten the patch release time by around 50 days.

Table 5: Results for model (2), effect of *Big_mshare*

Using as *Concentration*:	$-n$					
Big_mshare $= 1$ when the affected web browser's market share is:	≥ 0.40		≥ 0.45		≥ 0.50	
	OLS (coef.)	NB (IRR)	OLS (coef.)	NB (IRR)	OLS (coef.)	NB (IRR)
Concentration	-5.361**	0.932**	-5.496**	0.924***	-5.850**	0.914***
	(2.552)	(0.0265)	(2.561)	(0.0263)	(2.602)	(0.0260)
Big_mshare	42.45	5.939***	21.33	8.925***	34.01	22.02***
	(34.78)	(2.367)	(42.03)	(4.227)	(51.62)	(12.72)
Big_mshare#Concentration	8.942	1.298***	4.349	1.435***	6.998	1.738***
(interaction term)	(5.447)	(0.0847)	(7.376)	(0.123)	(9.489)	(0.190)
Vulnerability specific variables						
Soft. and vendor specific variables						
Disclosure effect variables	Coefficients of other variables available in Appendix					
Time effect variables						
Observations	586	586	586	586	586	586
R-squared	0.394		0.388		0.389	
Wald chi-squared		241.40		239.96		246.04

Robust standard errors in parentheses
*** p<0.01, ** p<0.05, * p<0.1
For OLS regressions coefficients are reported. For NB regressions IRR are reported.

Table 5 summarizes the regression results for model (2) which focuses on the change in the responsiveness of a publisher when it highly dominates the market. In this model, we only use the number of firms as a concentration measure. We do not test the model with the HHI because it is highly correlated to the explanatory variable of interest *Big_mshare*. In each pair of columns, we report the estimation results with different threshold values of market share for the *Big_mshare* dummy. That is, we tested the model with *Big_mshare* equal to 1 when the affected vendor's market share is more than 40, 45, or 50% of the market. As in Table 4, we report both OLS and NB regression results. We find that (1) firms release security patches less rapidly and (2) the positive effect of market concentration on their responsiveness is reduced, when they have a significant market share at the time the vulnerability is discovered. First, either the coefficient of Big_mshare is positive or its Incident Rate Ratio (IRR) is larger than 1, meaning that highly dominant firms release patches more slowly than when they do not have a significant market share. Moreover, in the case of NB estimations, the higher the threshold market share, the higher is the IRR. In other words, the higher the market share of the dominant firm, the larger its negative effect on the patch release time is. Secondly, the interaction term $Big\_mshare\#Concentration$ shows also a negative effect on the speed of patch release. That is, the positive effect of market concentration on reducing the patching time is weakened when the affected publisher has a significant market share. Moreover, this negative effect is strengthened with a higher threshold of market share for the Big_mshare dummy. For instance, the positive effect of market concentration is likely to be reduced by 44% ($1.738 - 1.298 = 0.44$) when firms have more than 50% of the market compared to when they have more than 40% of the market. Besides, we note that only estimates of negative binomial regressions are significant.

To check the robustness of our main findings, additional estimations are carried out. First, we exclude the vendor dummies and only include $open\_source$ dummy and $software\_age$ as software and vendor specific characteristics. The estimation results for our main explanatory variables are qualitatively identical (Table 8 in Appendix). Secondly, controlling for the data source of vulnerability reports does not affect the results (Table 9 in Appendix).[21] Also, not controlling for the vulnerability and software specific effects does not qualitatively affect our main results (Table 10 in Appendix).

9 Conclusion

In this paper, we have investigated the factors that influence the patching speed of web browser vulnerabilities. We have especially focused on the effect of market concentration and the high dominance of the affected firm, using a dataset of 586 web browser vulnerabilities identified and patched over a period of ten years.

Contrary to many empirical works examining the link between competition and quality, we find that market concentration is not necessarily harmful to quality provision. We explain this by a particularity of the market we study: because the software is provided free of charge to users and because companies derive their revenues from a neighboring market that makes use of the usage traffic, firms have more incentives to invest in quality when there are less competitors. Nevertheless, if we assume the existence of an installed base where loyal consumers keep their former choice despite the degradation of security quality, then the positive effect of market concentration is weakened: indeed, the impact of market concentration on quickening the patch release time is reduced when the affected web browser highly dominates the market.

[21] Our data come mainly from three vulnerability research programs, of which one program has been launched in 2014 while we study a 10-year period from 2007 to 2017

Academics and practitioners together have largely insisted on the danger of software monoculture. The basic reasoning is that using the same system in a connected network increases the interdependence between the *nodes* thus it makes the whole network more vulnerable to attacks (Böhme, 2005). But the reality is of course more complex (Schneier, 2010). Our results is complementary to this idea and suggest that it may be important as well to take account the vendors' actual incentives to provide a better security level.

Furthermore, in line with some practitioners' opinion, we find that diverse externalities such as the severity of the vulnerability, information disclosure or open source licensing also significantly impact the vendor's incentives to release a patch more quickly.

Appendix

Table 6: Description of variables

Variable	Description
Patching time	($Patching\_time$)
	Time spent by the editor to release a patch (unit: number of days)
Market concentration	($Concentration$)
$-n$	n represents the number of firms in the market at the time the vulnerability is discovered
HHI	The Herfindahl-Hirschman Index at the vulnerability discovery date
Significant market share	(Big_mshare)
	1 if the affected web browser publisher's market share at the moment the vulnerability is discovered is greater than 0.50% (0.40%) of the market, 0 otherwise
Vulnerability specific effects	($Vuln\_char$)
$vulnerability\_severity$	Vulnerability severity score (CVSS base score from 1 to 10)
vulnerabililty type	Dummies for type of weakness such as "cross site scripting", "SQL injection", "Information leak", etc. 12 type of weaknesses were associated to web browsers vulnerabilities.
Software and vendor specific effects	($V\&Soft\_char$)
$software\_age$	Age of the software at the time when the vulnerability is discovered (unit: number of years)
vendor dummies	Dummies that identifies the affected web browser publisher (Apple, Google, Mozilla. For Microsoft, Apple & Google & Mozilla = 0)
$open\_source$	1 if the affected web browser's engine is open source, 0 otherwise
Vulnerability information disclosure	($Disclosure$)
$earlier\_disclosure$	1 if the vulnerability was publicly disclosed earlier than the conventional period of time of 90 days after being notified to the vendor, 0 otherwise
Time effect	(T_effect)
$qyear$	The date on which the vulnerability was reported to the editor (unit: date in quarter)

Table 7: Summary statistics

Variable	Subdivision	Mean	Minimum	Maximum	Standard Deviation
$Patching\_time$		100.2	0	302	52.6
	Apple	100.5	0	302	64.3
	Google	66.5	1	282	76.2
	Microsoft	106.0	2	282	45.8
	Mozilla	73.8	1	240	58.4
$-n$		-6.8	-8	-4	1.02
	Apple	6.2	4	8	1.15
	Google	7	5	8	0.84
	Microsoft	6.9	4	8	0.95
	Mozilla	6.3	4	8	1.1
HHI		0.309	0.246	0.575	0.639
	Apple	0.343	0.246	0.575	0.638
	Google	0.299	0.246	0.458	0.431
	Microsoft	0.302	0.246	0.575	0.606
	Mozilla	0.340	0.246	0.575	0.790
Big_mshare	($m.s. \geq 0.40$)	0.13	0	1	0.33
Big_mshare	($m.s. \geq 0.50$)	0.09	0	1	0.28
$vulnerability\_severity$		8.75	2.6	10	1.30
$software\_age$		5.98	0	14.8	4.18
$open\_source$		0.26	0	1	0.44
$earlier\_disclosure$		0.53	0	1	0.50
$qyear$		2013q2	2007q1	2016q4	8.7

Number of observation: 586

Table 8: Detailed results for Table 5

Using as *Concentration*:	-*n*					
Big_mshare = 1 when the affected web browser's market share is:	≥ 0.40		≥ 0.45		≥ 0.50	
	OLS (coef.)	NB (IRR)	OLS (coef.)	NB (IRR)	OLS (coef.)	NB (IRR)
Concentration	-5.361**	0.932**	-5.496**	0.924***	-5.850**	0.914***
	(2.552)	(0.0265)	(2.561)	(0.0263)	(2.602)	(0.0260)
Big_mshare	42.45	5.939***	21.33	8.925***	34.01	22.02***
	(34.78)	(2.367)	(42.03)	(4.227)	(51.62)	(12.72)
Big_mshare#Concentration	8.942	1.298***	4.349	1.435***	6.998	1.738***
(interaction term)	(5.447)	(0.0847)	(7.376)	(0.123)	(9.489)	(0.190)
Vulnerability specific effects						
vulnerability_severity	-4.826**	1.069***	-4.999**	1.062**	-4.896**	1.060**
	(2.031)	(0.0259)	(2.040)	(0.0259)	(2.068)	(0.0266)
vulnerability_type dummies						
*cwe*119	-8.377	0.896	-7.735	0.933	-9.738	0.950
	(10.63)	(0.0992)	(10.57)	(0.103)	(10.61)	(0.106)
*cwe*17	-30.40	0.468*	-24.46	0.745	-24.41	0.607
	(29.43)	(0.192)	(31.46)	(0.312)	(32.64)	(0.248)
*cwe*94	19.42	1.450***	19.84	1.497***	15.78	1.487***
	(14.99)	(0.189)	(14.91)	(0.195)	(14.91)	(0.196)
*cwe*200	-21.03	1.202	-21.34	1.203	-20.13	1.265
	(14.16)	(0.252)	(14.27)	(0.254)	(14.84)	(0.269)
*cwe*20	-9.467	1.041	-8.835	1.095	-9.109	1.124
	(12.50)	(0.150)	(12.47)	(0.157)	(12.23)	(0.161)
*cwe*189	-25.05	0.745	-22.66	0.830	-25.95	0.824
	(16.12)	(0.146)	(15.59)	(0.161)	(16.07)	(0.161)
*cwe*254security	-10.85	1.268	-10.86	1.282	-12.04	1.290
	(13.71)	(0.689)	(13.74)	(0.700)	(13.96)	(0.704)
*cwe*22	-73.81***	0.0127***	-72.09***	0.0147***	-53.22***	0.0213***
	(19.69)	(0.0144)	(18.93)	(0.0167)	(12.55)	(0.0242)
*cwe*264	-18.69	1.114	-18.40	1.140	-20.28	1.152
	(14.09)	(0.182)	(14.07)	(0.187)	(14.17)	(0.188)
*cwe*362	-24.13	0.841	-25.95	0.801	-25.38	0.772
	(17.02)	(0.345)	(18.29)	(0.331)	(18.05)	(0.320)
*cwe*399	6.765	1.217*	7.690	1.303**	4.015	1.316**
	(11.71)	(0.142)	(11.59)	(0.149)	(11.68)	(0.154)
*cwe*704	5.209	2.012	-0.201	1.335	8.935	1.015
	(20.51)	(1.157)	(17.59)	(0.751)	(14.43)	(0.556)
Soft. and vendor specific effects						
software_age	0.768*	1.005	0.767*	1.005	0.855*	1.004
	(0.442)	(0.00571)	(0.442)	(0.00577)	(0.449)	(0.00576)
apple	-9.522	1.246***	-9.829	1.242***	-11.71	1.263***
	(7.422)	(0.0969)	(7.423)	(0.0972)	(7.720)	(0.0992)
google	-18.24	1.089	-19.77	0.966	-13.06	0.910
	(17.30)	(0.140)	(16.57)	(0.115)	(14.96)	(0.104)
mozilla	-23.42**	1.006	-23.94***	0.996	-25.53***	1.020
	(9.179)	(0.0907)	(9.177)	(0.0905)	(9.504)	(0.0930)
Disclosure effect						
earlier_disclosure	-48.72***	0.601***	-48.64***	0.604***	-48.27***	0.605***
	(3.557)	(0.0274)	(3.565)	(0.0277)	(3.580)	(0.0277)
Time effect						
qyear	-1.451***	1.017***	-1.479***	1.017***	-1.630***	1.017***
	(0.345)	(0.00133)	(0.346)	(0.00134)	(0.392)	(0.00135)
Constant	444.2***	0.262***	450.2***	0.265***	482.0***	0.265***
	(88.49)	(0.0162)	(88.90)	(0.0164)	(99.09)	(0.0164)
Observations	586	586	586	586	586	586
R-squared	0.394		0.388		0.389	
Wald chi-squared		241.40		239.96		246.04

Robust standard errors in parentheses
*** p<0.01, ** p<0.05, * p<0.1
For OLS regressions coefficients are reported. For NB regressions AME or IRR are reported.

Table 9: Estimation results with *open_source* variable

Using as *Concentration*:	-*n*		HHI		-*n*	
Big_mshare = 1 when the affected web browser's market share is:					$\geq$ 0.50	
	OLS (coef.)	NB (AME)	OLS (coef.)	NB (AME)	OLS (coef.)	NB (IRR)
Concentration	-5.142**	-4.129*	-81.99*	-102.0**	-5.016*	0.935**
	(2.476)	(2.293)	(43.28)	(41.79)	(2.668)	(0.0256)
Big_mshare					33.36	23.13***
					(46.37)	(12.86)
Big_mshare#Concentration					7.092	1.757***
(interaction term)					(7.722)	(0.180)
Vulnerability specific effects						
vulnerability_severity	-5.568***	-5.239**	-6.049***	-6.086**	-5.333**	1.066***
	(2.126)	(2.520)	(2.279)	(2.549)	(2.162)	(0.0248)
vulnerability_type dummies						
*cwe*119	-5.928	-2.201	-6.264	-1.533	-8.083	0.906
	(10.09)	(10.51)	(10.07)	(10.48)	(10.47)	(0.1000)
*cwe*17	-32.93	-41.59*	-32.54	-40.54*	-36.14	0.447**
	(30.11)	(22.23)	(30.02)	(22.57)	(30.46)	(0.178)
*cwe*94	19.77	22.65	18.26	21.75	18.58	1.457***
	(14.74)	(15.01)	(14.56)	(14.84)	(14.96)	(0.190)
*cwe*200	-24.02*	-18.19	-27.04*	-20.27	-23.88*	1.182
	(14.10)	(16.91)	(14.45)	(16.42)	(14.12)	(0.249)
*cwe*20	-7.878	0.220	-8.327	-0.200	-9.969	1.042
	(12.23)	(13.76)	(12.24)	(13.68)	(12.55)	(0.149)
*cwe*189	-25.71*	-24.61*	-25.28*	-22.49	-28.69*	0.696*
	(15.31)	(13.88)	(15.19)	(14.23)	(15.63)	(0.135)
*cwe*254	-10.85	-4.694	-10.81	-3.863	-12.39	1.275
	(13.59)	(49.48)	(13.56)	(49.79)	(13.88)	(0.694)
*cwe*22	-73.19***	-99.01***	-71.13***	-98.99***	-76.19***	0.0123***
	(11.68)	(2.743)	(11.42)	(2.751)	(12.70)	(0.0139)
*cwe*264	-19.05	-16.81	-20.69	-17.66	-20.99	1.097
	(13.90)	(13.29)	(14.06)	(13.08)	(14.27)	(0.178)
*cwe*362	-35.75*	-44.02**	-38.47**	-45.74**	-30.45**	0.751
	(18.87)	(21.89)	(17.46)	(21.13)	(14.42)	(0.305)
*cwe*399	8.023	15.30	7.415	16.18	5.712	1.209
	(11.42)	(11.78)	(11.37)	(11.80)	(11.80)	(0.140)
*cwe*704	8.962	18.33	10.84	23.47	21.74**	1.817
	(11.10)	(60.61)	(10.85)	(63.14)	(9.682)	(1.022)
Soft. and vendor specific effects						
software_age	0.874*	1.022**	0.895**	1.030**	0.859*	1.007
	(0.447)	(0.520)	(0.451)	(0.518)	(0.455)	(0.00559)
open_source	-17.53***	-22.46***	-18.40***	-23.64***	-16.10**	1.140**
	(5.352)	(5.327)	(5.295)	(5.253)	(6.466)	(0.0688)
Disclosure effect						
earlier_disclosure	-49.59***	-49.56***	-49.87***	-50.19***	-49.64***	0.593***
	(3.563)	(4.449)	(3.580)	(4.465)	(3.545)	(0.0268)
Time effect						
qyear	-1.544***	-1.410***	-1.644***	-1.649***	-1.480***	1.017***
	(0.283)	(0.348)	(0.305)	(0.375)	(0.339)	(0.00134)
Constant	470.7***		557.3***		457.6***	0.264***
	(70.56)		(89.99)		(87.49)	(0.0163)
Observations	586	586	586	586	586	586
R-squared	0.384		0.383		0.386	
Wald chi-squared		232.37		235.06		239.15

Robust standard errors in parentheses
*** p<0.01, ** p<0.05, * p<0.1
For OLS regressions coefficients are reported. For NB regressions IRR are reported.

Table 10: Estimation results without vulnerability and software/vendor characteristics control variables

Using as *Concentration*:	*-n*		HHI		*-n*	
Big_mshare = 1 when the affected web browser's market share is:					≥ 0.50	
	OLS (coef.)	NB (AME)	OLS (coef.)	NB (AME)	OLS (coef.)	NB (IRR)
Concentration	-5.039**	-5.475**	-74.94*	-106.0**	-7.138***	0.931***
	(2.443)	(2.339)	(40.02)	(41.33)	(2.664)	(0.0252)
Big_mshare					122.5***	54.18***
					(35.67)	(28.31)
Big_mshare#Concentration					20.72***	1.974***
					(5.760)	(0.192)
Disclosure effect						
earlier_disclosure	-53.84***	-53.88***	-54.19***	-54.73***	-51.41***	0.621***
	(3.588)	(4.596)	(3.560)	(4.629)	(3.572)	(0.0299)
Time effect						
qyear	-1.536***	-1.504***	-1.569***	-1.640***	-1.250***	1.020***
	(0.248)	(0.279)	(0.264)	(0.301)	(0.266)	(0.000914)
Constant	422.6***		487.2***		344.6***	0.320***
	(46.77)		(65.14)		(55.01)	(0.0193)
Observations	586	586	586	586	586	586
R-squared	0.325		0.323		0.336	
Wald chi-squared		167.69		168.76		176.04

Robust standard errors in parentheses
*** p<0.01, ** p<0.05, * p<0.1
For OLS regressions coefficients are reported. For NB regressions AME or IRR are reported.

Table 11: Regression of *vulnerability_severity* by *vulnerability_type* dummies

dependent var:	*vulnerability_severity*
*cwe*119	0.920***
	(0.160)
*cwe*17	-0.887
	(0.648)
*cwe*94	1.214***
	(0.194)
*cwe*200	-3.776***
	(0.314)
*cwe*20	0.585**
	(0.228)
*cwe*189	1.184***
	(0.302)
*cwe*254	-3.737***
	(0.904)
*cwe*22	-0.537
	(0.904)
*cwe*264	-2.324***
	(0.243)
*cwe*362	-2.987***
	(0.648)
*cwe*399	1.169***
	(0.167)
*cwe*704	-1.237
	(0.904)
Constant	8.037***
	(0.151)
Observations	586
R-squared	0.541

Standard errors in parentheses
*** p<0.01, ** p<0.05, * p<0.1

References

F. Allen. Reputation and product quality. *The RAND Journal of Economics*, pages 311–327, 1984.

V. Anand and D. Saxena. Comparative study of modern web browsers based on their performance and evolution. In *Computational Intelligence and Computing Research (ICCIC), 2013 IEEE International Conference on*, pages 1–5. IEEE, 2013.

R. Anderson and T. Moore. Information security economics–and beyond. pages 68–91, 2007.

C. Argenton and J. Prüfer. Search engine competition with network externalities. *Journal of Competition Law and Economics*, 8(1):73–105, 2012.

A. Arora, J. P. Caulkins, and R. Telang. Research note-sell first, fix later: Impact of patching on software quality. *Management Science*, 52(3):465–471, 2006a.

A. Arora, A. Nandkumar, and R. Telang. Does information security attack frequency increase with vulnerability disclosure? an empirical analysis. *Information Systems Frontiers*, 8(5):350–362, 2006b.

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

A. Arora, C. Forman, A. Nandkumar, and R. Telang. Competition and patching of security vulnerabilities: An empirical analysis. *Information Economics and Policy*, 22(2):164–177, 2010a.

A. Arora, R. Krishnan, R. Telang, and Y. Yang. An empirical analysis of software vendors' patch release behavior: impact of vulnerability disclosure. *Information Systems Research*, 21(1):115–132, 2010b.

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

R. D. Banker, I. Khosla, and K. K. Sinha. Quality and competition. *Management Science*, pages 1179–1192, 1998.

Y. Beres, J. Griffin, S. Shiu, M. Heitman, D. Markle, and P. Ventura. Analysing the performance of security solutions to reduce vulnerability exposure window. pages 33–42, 2008.

R. Böhme. Cyber-insurance revisited. In *WEIS*, 2005.

R. H. Bork and J. G. Sidak. What does the chicago school teach about internet search and the antitrust treatment of google? *Journal of Competition Law and Economics*, 8(4):663–700, 2012.

K. Campbell, L. A. Gordon, M. P. Loeb, and L. Zhou. The economic cost of publicly announced information security breaches: empirical evidence from the stock market. *Journal of Computer Security*, 11(3):431–448, 2003.

H. Cavusoglu, H. Cavusoglu, and J. Zhang. Security patch management: Share the burden or share the damage? *Management Science*, 54(4):657–670, 2008.

C. Chambers, P. Kouvelis, and J. Semple. Quality-based competition, profitability, and variable costs. *Management Science*, 52(12):1884–1895, 2006.

J. P. Choi, C. Fershtman, and N. Gandal. Network security: Vulnerabilities and disclosure policy. *The Journal of Industrial Economics*, 58(4):868–894, 2010.

A. M. Cohen and M. J. Mazzeo. Competition, product differentiation and quality provision: an empirical equilibrium analysis of bank branching decisions. 2004.

S. Domberger and A. Sherr. The impact of competition on pricing and quality of legal services. *International Review of Law and Economics*, 9(1):41–56, 1989.

N. Economides. Quality variations in the circular model of variety-differentiated products. *Regional Science and Urban Economics*, 23(2):235–257, 1993.

D. S. Evans. Some empirical aspects of multi-sided platform industries. *Review of Network Economics*, 2(3), 2003.

S. Frei, M. May, U. Fiedler, and B. Plattner. Large-scale vulnerability analysis. pages 131–138, 2006.

E. Gal-Or. Quality and quantity competition. *The Bell Journal of Economics*, pages 590–600, 1983.

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

L. A. Gordon, M. P. Loeb, W. Lucyshyn, and T. Sohail. The impact of the sarbanes-oxley act on the corporate disclosures of information security activities. *Journal of Accounting and Public Policy*, 25(5):503–530, 2006.

C. Ioannidis, D. Pym, and J. Williams. Information security trade-offs and optimal patching policies. *European Journal of Operational Research*, 216(2):434–444, 2012.

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

B. C. Kim, P.-Y. Chen, and T. Mukhopadhyay. An economic analysis of the software market with a risk-sharing mechanism. *International Journal of Electronic Commerce*, 14(2):7–40, 2009.

P. Klemperer. Markets with consumer switching costs. *The quarterly journal of economics*, 102(2):375–394, 1987.

J. Lerner and J. Tirole. Some simple economics of open source. *The journal of industrial economics*, 50(2): 197–234, 2002.

M. J. Mazzeo. Product choice and oligopoly market structure. *The RAND Journal of Economics*, pages 221–242, 2002.

M. J. Mazzeo. Competition and service quality in the us airline industry. *Review of industrial Organization*, 22 (4):275–296, 2003.

G. McGraw and C. CTO. Exploiting software: How to break code. In *Invited Talk, Usenix Security Symposium, San Diego*, 2004.

Monopolkommission. Competition policy: The challenge of digital markets, 2015.

R. Narasimhan, S. Ghosh, and D. Mendez. A dynamic model of product quality and pricing decisions on. *Decision Sciences*, 24(5):893, 1993.

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

S. Ransbotham, S. Mitra, and J. Ramsey. Are markets for vulnerabilities effective? *ICIS 2008 Proceedings*, page 24, 2008.

J.-C. Rochet and J. Tirole. Two-sided markets: a progress report. *The RAND journal of economics*, 37(3): 645–667, 2006.

U. Ronnen. Minimum quality standards, fixed costs, and competition. *The RAND Journal of economics*, pages 490–504, 1991.

B. Schneier. Managed security monitoring: Closing the window of exposure. *Counterpane Internet Security*, 2000.

B. Schneier. The dangers of a software monoculture. *Information Security Magazine*, 2010.

A. Shostack. Quantifying patch management. *Secure Business Quarterly*, 3(2):1–4, 2003.

F. Smedberg. Performance analysis of javascript. 2010.

A. M. Spence. Monopoly, quality, and regulation. *The Bell Journal of Economics*, pages 417–429, 1975.

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

H. R. Varian. High-technology industries and market structure. *University of California, Berkeley*, 33, 2001.

D. Waterman. Diversity and quality of information products in a monopolistically competitive industry. *Information Economics and Policy*, 4(4):291–303, 1990.