An Empirical Analysis of Vendor Response to Disclosure Policy

Ashish Arora, Ramayya Krishnan, Rahul Telang, Yubao Yang*
{Ashish, rk2x, rtelang, yubaoy}@andrew.cmu.edu

H. John Heinz III School of Public Policy and Management
Carnegie Mellon University, Pittsburgh PA 15213

This version: March 4, 2005

Abstract

Software vulnerability disclosure has generated intense interest and debate. In particular, there have been arguments made both in opposition to and in favor of alternatives such as full and instant disclosure and limited or no disclosure. An important consideration in this debate is the behavior of the software vendor. Does vulnerability disclosure policy have an effect on patch release behavior of software vendors? This paper compiles a unique data set from CERT/CC and Security Focus databases to answer this question. Our results suggest that early disclosure has significant positive impact on the vendor patching speed. Open source vendors patch more quickly than closed source vendors and severe vulnerabilities are patched faster. We also find that vendors respond slower to vulnerabilities not disclosed by CERT/CC. This might reflect unmeasured differences in the severity and importance of vulnerabilities. It might also reflect the stronger lines of communication between CERT/CC and vendors, and the value of the vulnerability analysis by CERT/CC. We also find that vendors are more responsible after the 9/11 event.

Keywords: Security vulnerability, disclosure policy, patching speed

* Corresponding author. Please send questions and comments to yubaoy@andrew.cmu.edu

1. Introduction

There is a contentious ongoing debate about when (and how) vulnerability information should be made public. While information about vulnerabilities enables some users to take precautions that prevent or reduce cyber security breaches, vulnerability information, especially when not accompanied by patches or workarounds, can benefit attackers more than users.

Sources that report vulnerability information range from federally funded quasi-government organizations like CERT/CC [1] to open-source online communities like SecurityFocus [2]. Traditionally, CERT/CC has been a key player in the domain of vulnerability disclosure. A typical sequence of events in the case of CERT/CC is as follows. A *benign identifier* reports the vulnerability to CERT/CC, which then researches the vulnerability to determine the severity level of this vulnerability. For severe vulnerability, CERT/CC contacts the vendors that they believe to be involved and provides them a certain time window (normally 45 days) to patch the vulnerability. After that time CERT/CC sends out public "vulnerability notes" warning users about the vulnerability. The vulnerability notes include links to any available patches and provide enough technical information about the vulnerability to enable users to take protective action.

As one of the most popular computer security vulnerability forums on internet, SecurityFocus is well known for its instant disclosure policy. SecurityFocus collects and publishes security vulnerability information through its full disclosure mailing list, called Bugtrag. Thus the security vulnerability information, once sent to Bugtrag by identifiers, is publicly available

[1] Established in 1988, the CERT® Coordination Center (CERT/CC) is a center of Internet security expertise, located at the Software Engineering Institute, a federally funded research and development center operated by Carnegie Mellon University.(See www.cert.org for more detail)

[2] SecurityFocus is one of the most comprehensive and trusted source of security information on the Internet. By the full disclosure mailing list Bugtrag, SecirtyFocus provides objective, timely and comprehensive security information to all members of the security community. (See www.securityfocus.com for more detail)

instantaneously. Unlike CERT/CC, which provides only enough technical information about the vulnerability to enable users to take protective action, SecurityFocus may publish technical details (including exploit code) of a vulnerability even if the vendor has not provided a patch.

While the proponents of instant disclosure claim that disclosing vulnerability information provides an impetus to the vendor to release patches early, the proponents of CERT/CC argue that *early disclosure* leaves users defenseless against attackers who can exploit the disclosed vulnerability and therefore, is socially undesirable (see Elias 2001 and Farrow 2000)[3]. Arora, Telang and Xu [2004] develop a theoretical model which implies that although early disclosure is not necessarily socially optimal, it would result in the vendor releasing a patch more quickly.

Further understanding of the implications of an optimal disclosure policy requires knowledge about the trade-offs that disclosure policy imposes on the three major participants who are affected by it. The first participant is the software vendor, who bears the cost of developing and testing a patch, and may suffer the loss of customer goodwill, reputation and even market share, when a vulnerability in their products is disclosed. The second participant is the user of the vulnerable system, who incurs the cost of installing a patch and may also suffer a variety of losses from security breaches and must also incur the cost of implementing the patch. The third set of participants is the black-hat hacker. Disclosure policies affect each participant differently and sometimes in a contradictory manner.

While recent theoretical work provides a theoretical model to understand how disclosure policies would affect these participants (see Arora, Telang and Xu 2004 for details), these predictions need to be empirically tested. A major obstacle has been lack of reliable data. Gordon et al.

[3]See also the debate between Robert Graham and Bruce Schneider. http://www.robertgraham.com/diary/disclosure.html

[1999] also acknowledge the importance of these issues, and point out the lack of hard evidence to assess the impact of various forms of vulnerability disclosure.

In this paper, we construct a unique data set from CERT/CC and SecurityFocus and publicly available data on vendors. Our datasets contains the vulnerability information[4] including when the vendor was notified, if and when a patch was made available,[5] if and when the vulnerability was publicly disclosed, patching time, the nature and severity of the vulnerability and the vendor information. All the vulnerabilities in the dataset are published within the time period from September 26, 2000 to August 11, 2003.

The rest of the paper is organized as follows. Section 2 provides a brief description of different vulnerability disclosure policies and the vendors' patching cost based on a software life cycle model; Section 3 sets out the economic model of the vendors' patching decision; Section 4 provides a description of the data and variables; Section 5 presents the estimation results with discussion. We conclude and discuss limitations of this research in section 6.

2. Vulnerability disclosure and vendor patching cost

A typical life cycle of vulnerability is shown in figure 3.1 (Arbaugh et al., 2000).

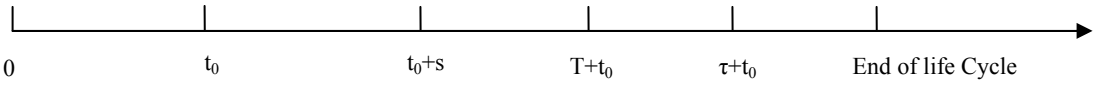

Figure 3.1 Software Life Cycles

[4] Note that a vulnerability may affect more than one vendor's product depend on the nature of the software. For example, BIND (the Berkeley Internet Name Domain) is a server application providing name resolution services on the Internet that is widely integrated by major UNIX/ Linux vendors. Buffer overflow vulnerability in BIND 8.2.x (published in 2001) allows remote intruders to gain access to systems, affects almost all the major UNIX/Linux servers running BIND 8.

[5] A software patch can be an upgrade (adding increased features), a bug fix, and a new hardware driver or update to address new issues such as security or stability problems. Most patches are free to download. But in come cases, the customers need to purchase the newer version at a discounted upgraded price (and requiring a reinstallation of the program) to have the vulnerability fixed. (www.softwarepatch.com)

At time '0' the product is released by a vendor and used by users. A vulnerability is discovered and reported to public sources, which in our case are CERT/CC or SecurityFocus, by a benign user[6]. If the vulnerability is reported to CERT/CC, CERT/CC will research the vulnerability and determine its severity level, which normally takes very short time. If the vulnerability is determined to be severe enough, CERT/CC will then contact vendors and provide them with a window of 45 days before disclosing it. If the vulnerability is reported to SecurityFocus, SecurityFocus will make it public immediately. In both case, vendors are notified at time t_0.

Disclosure policy T requires that this vulnerability is kept secret until time $t_0 + T$ and disclosed after that. So generally the expected disclosure time T is 45 for CERT/CC and 0 for SecurityFocus. Vendors provide a patch for this vulnerability at a calendar time $\tau + t_0$. Attackers can find and exploit the vulnerability at time $\mathbf{t_0+s}$[7].

Arora, Telang and Xu [2003] specify the vendor patch decision problem and analyze how the time to develop a patch changes with T, the time to disclosure. A key implication of their model is that though reducing T generally reduces the time to patch, instant disclosure is typically not optimal because developing, testing and installing patches is costly and time consuming.

In this paper we focus on testing the hypothesis that vendors respond more quickly to early disclosure. Since SecurityFocus discloses vulnerability more quickly, we also expect to see that vendors respond faster to SecurityFocus. However, empirical testing is complicated because vulnerabilities not disclosed by CERT are likely to be less severe. In addition, CERT disclosures

[6] A benign user is one not interested in exploiting this vulnerability. Also note that if an attacker discovers the vulnerability first then it will immediately attack and any disclosure policy is a moot. And if a vendor discovers the vulnerability first, then in almost all cases they will keep the vulnerability secrete or publish it till they have a fix.

[7] A vulnerability can be detected by a hostile attacker first. In this case $s < 0$, and CERT/CC or Security would find out when there is an attack at t_0.

may elicit more attention from the vendors because of CERT/CC's salience or because CERT staff may have good lines of communication with the vendor.

3. Data and variables

The vulnerabilities studied in our research are from the vulnerability publications of CERT/CC and SecurityFocus. We collected information of whether the vendors released a patch and how much time they took for each vulnerability in our sample. We further augmented the data by adding the vendor characteristics, which are colleted from the vendor website or from the Hoover's online database, and vulnerability characteristics, which are colleted from ICAT database. The ICAT database (http://icat.nist.gov) is a product of the Computer Security Division at the National Institute of Standards and Technology (NIST). NIST assigns unique identifiers known as CVE ID to the major vulnerabilities that are known publicly and records the standard technical information of the vulnerability as well as the related software. CERT/CC and SecurityFocus are the two most important information sources of ICAT database. Over 60% of the vulnerabilities in the ICAT database are published by either CERT/CC or SecurityFocus.

A typical sequence of events in case of CERT/CC is as follows. A *benign identifier* reports the vulnerability to CERT/CC. CERT/CC does quick preliminary research on the vulnerability and then contacts all the suspect vendors if the research shows that the vulnerability is new and severe. CERT/CC gets on average over 3000 vulnerability reports from the public each year, and they process and publish about 10% of these reports. For the vulnerabilities that they publish, CERT/CC coordinates with vendors on patch development and typically allows them a 45 days time window. After 45 days have elapsed, CERT publishes the vulnerability in the form of "vulnerability notes". The vulnerability notes provide enough technical information about the

vulnerability to enable users to take protective action. All the vendors that are previously notified by CERT/CC would be included in the vulnerability notes as well.

For the time period from 9/26/2000 to 8/11/2003 that we studied, CERT/CC published 526 vulnerabilities notes. 622 vendors are disclosed in these vulnerability notes. To be able to use the standard technical information of the vulnerabilities in our analysis, we include only the 494 vulnerabilities that are documented by NIST ICAT database in our sample. Of these 494 vulnerabilities, 416 vulnerabilities are also published by SecurityFocus and 78 vulnerabilities are published only through CERT/CC.

In order to obtain a sample of vulnerabilities that are published in SecurityFocus alone, we proceeded as follows. SecurityFocus publishes vulnerabilities that are reported to or discussed on the Bugtrag mailing list. Between from 9-26-2000 to 8-11-2003, SecurityFocus published 2791 NIST ICAT database documented vulnerabilities. Of these 2407 are published by SecurityFocus alone (not published by CERT/CC). We randomly picked 131 out of these 2407 vulnerabilities.

Though SecurityFocus tends to be associated with early disclosure, not all vulnerabilities published by SecurityFocus are instantly disclosed. Sometime the identifier might have informed the vendor before reporting to SecurityFocus. The true vendor notification date can be acquired by going through the original document of the interaction between vendors and identifiers, which is publicly available on SecurityFocus website. In our SecurityFocus sample, there are about 30% observations that vendors were pre-informed by the identifiers.

This paper focuses on (1) the time elapsed between notification and patch availability, (2) how this changes with whether the vulnerability is disclosed early or not. We divide the sample into following scenarios.

Scenario I: $t_0 < t_0 + \tau <= t_0 + T$

In this scenario, the vendor is notified and they have patch available before the vulnerability getting disclosed.

Scenario II: $t_0 < t_0 + T < t_0 + \tau$

In this scenario, the vendors are notified. However, before they have the patch ready, someone discloses this information (for example, CERT notifies the vendors but SecurityFocus might disclosure it before the patch is ready). Observations in this scenario and T < 10 days would be treated as early disclosure. Various sensitivity tests around T < 10 can be performed.

***Scenario III*:** T=0

This is the scenario of instant disclosure. Vendors are notified at the same time when the vulnerabilities are disclosed. Vulnerabilities in this scenario are treated as early disclosure. Scenarios II and III allows us to compare the vendor behavior from scenario I and test the impact of early disclosure

***Scenario IV*:** $\tau = 0$

This scenario normally happen when the vulnerabilities are first disclosed by the vendors. Instead of being notified, the vendors notify CERT/CC and/or SecurityFocus about the vulnerability along with the patch. Vulnerabilities in this scenario add no value to our analysis and are excluded from our sample.

Variables

The sample studied in this paper is the vulnerabilities from scenario I - III. The final analysis data set contains 1280 observations, related to 275 vendors and 305 vulnerabilities[8].

[8] A vendor may exposed to multiple vulnerabilities and a vulnerabilities may affect multiple vendors.

Vendor's patching speed is measured by the number of days elapsed between notification and availability of the patch. In the case vendors are not notified or notified after publication, the patching time is measured by the number of days elapsed between publication and availability of the patch. Figure 3-1 provides the distribution of patching speed. Note that we only include the observations that are patched in one year in the figure. The observations that are patched in more than one year counts about 2.8% of the total observations.

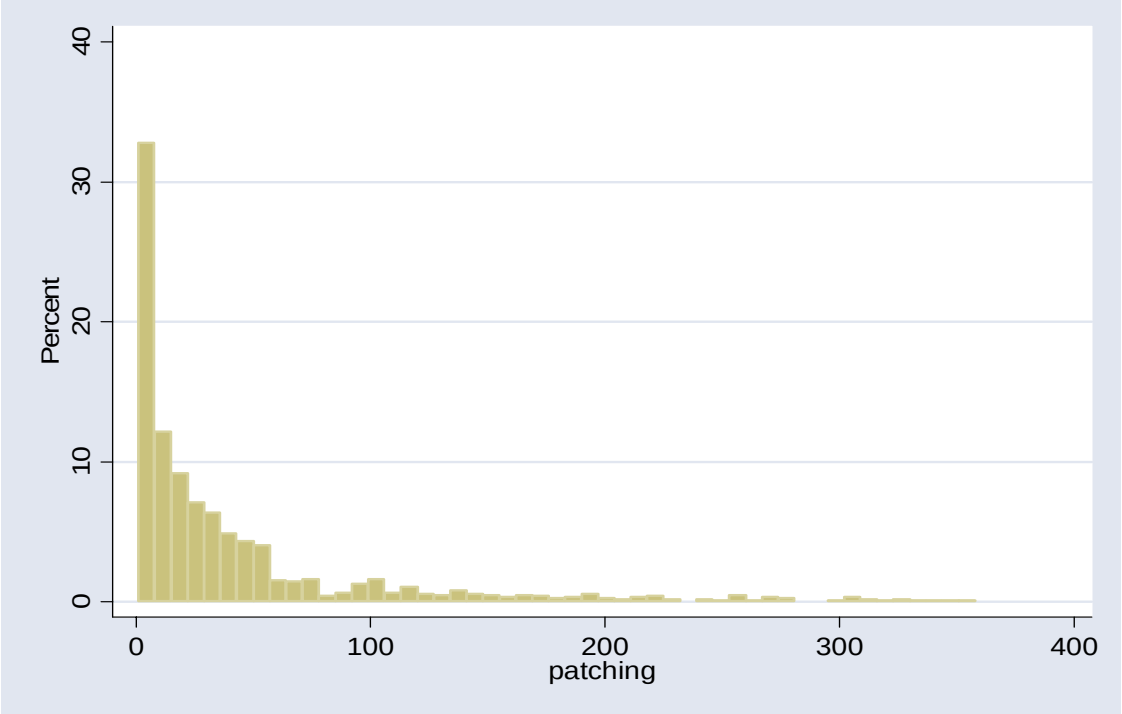

Figure 3-1 the distribution of patching time

The x axis is the patching time by number of days. The y axis is the probability of the patching time within certain range. Each bar in the figure represent a time period of one week.

Early disclosure is said to take place if the vulnerability is disclosed within 10 days of notification before vendors fixes the vulnerability. Instant disclosure is the special case of early disclosure, in which the vulnerability is disclosed with vendors not notified in advance. We use

a dummy variable ***Early*** indicating if the vulnerability is disclosed early to capture the early disclosure effect.

We use another dummy variable CERT, which indicating if the vulnerability is published by CERT/CC to capture the effect of CERT/CC. Moreover, we allow the interaction of CERT effect and early disclosure effect by construct three dummy variables. Table 4-1 provides a list of variable constructions with their means.

Table 4-1 Variables definitions and means

Variable	Definition	Mean	Std. Dev.
Early	=1 for the observations that are instantly disclosed or disclosed within 10 days of notification while patch comes after the disclosure	0.69	0.46
CERT	=1 for the observations published by CERT/CC	0.86	0.35
SecurityFocus	=1 for the observations published by SecurityFocus	0.94	0.24
CERT_alone	=1 for the observations that are published by CERT/CC alone		
Early_by_CERT	=1 for the observations jointly published by CERT/CC and SecurityFocus and early disclosed by CERT/CC	0.06	0.24
Early_by_SF	=1 for the observations jointly published by CERT/CC and SecurityFocus and early disclosed by SecurityFocus	0.41	0.49
Notify	=1 for the observations that vendors are notified about the vulnerability by CERT/CC or other organizations.	0.72	0.45

The variables controlling for vendor characteristics are discussed in Table 4-2, which include size measured as the number of employees, firm type dummy (public firm or not), software license type dummy (open source or not) and a dummy variable indicating if we have been able to find the vendor information. The vendors that we have been unable to locate mostly are either foreign companies or very small organizations.

Table 4-2 vendor characteristics (N=275)

Variable description	Mean	Std. Dev.	Min	Max
Vendor employee size	12173.67	55331.11	0	517887
Public firm	0.22	0.41	0	1
Open source software	0.11	0.32	0	1
Vendors with no information available	0.51	0.50	0	1

We use the CERT/CC vulnerability metric as our measurement of vulnerability severity, which is a number between 0 and 180[9]. Vulnerabilities published by SecurityFocus alone do not have a CERT/CC metric. To solve this problem, we first use the vulnerability jointly published by CERT/CC and SecurityFocus to create linear projection of log(metric) on the vulnerability technical characteristics. Then we fit metric value for the vulnerability published by SecurityFocus alone using this projection. Table 4-3 summarized the severity of the vulnerability measured by CERT/CC vulnerability metric.

Table 4-3 Vulnerability severity metric

Variable description	*Mean*	*Std. Dev.*	*Min*	*Max*
Vulnerability severity metric	16.16	17.59	0	108.16

We also control for the impact of September 11 events on the software vendors. About 53% of the vulnerabilities included in our sample are published after September 11th 2001.

The vulnerability characteristic dummies are discussed in Table 4-3.

[9] See http://www.kb.cert.org/vuls/html/fieldhelp#metric for definition of CERT vulnerability Metric.

Table 4-3 vulnerability characteristics (N=305)

Variable	Description	Mean	Std. Dev.
AR_LR	Remote attachable. Attacks that utilize the vulnerability can be launched across a network against a system without the user having previous access to the system.	0.76	0.43
LT_OAP	Obtain all privilege. Vulnerabilities enable an attack that can gain all the privilege of a system.	0.25	0.43
LT_C	Lose Confidentiality. Vulnerabilities enable an attack that can directly steal information from a system.	0.14	0.35
LT_I	Lose Integrity. The vulnerability enables an attack that can directly change the information residing on or passing through a system.	0.17	0.37
VT_IVE	Boundary overflow. Vulnerabilities, when the input being received by a system, be it human or machine generated, cause the system to exceed an assumed boundary thereby causing a vulnerability.	0.54	0.50
VT_BO	Buffer overflow. Vulnerabilities caused by input being received by a system that is longer than the expected input length. If the system does not check for this condition then the input buffer fills up and overflows the memory allocated for the input. By cleverly constructing this extra input, an attacker can cause the system to crash or even to execute instructions on behalf of the attacker.	0.30	0.46
VT_ECE	Exceptional condition handling error. Vulnerabilities caused by an exceptional condition that has arisen. The handling (or mishandling) of the exception by the system enables a vulnerability.	0.10	0.30
VT_DE	Design Error. Vulnerability is characterized as a "Design error" if there are no errors in the implementation or configuration of a system, but the initial design causes a vulnerability to exist.	0.23	0.42
EC_SA	Server Application. Vulnerabilities occurred in an application providing services to the other users or computers on the network.	0.56	0.50

4. Empirical evidence

4.1 Methodology

There are very few observations in the data set which the patching time is unusually large. To avoid the result being biased by these extreme values, we adopt a log linear regression model. We express the log patching time associated with a particular disclosure policy, vendor and vulnerability type as:

$$Log(Patching\_time_j^i) = \mathrm{B}_0 + \mathrm{B}_1 disclosure\_policy_j^i + \mathrm{B}_2 vendor\_characteristics^i + \mathrm{B}_3 Vul\_characteristics_j + \varepsilon_j^i \quad (1)$$

The superscript i indexes the vendors while the subscript j indexes the vulnerabilities. B_0 is the average patching time without taking into consideration of the policy effect, vendor characteristics as well as the vulnerability natures. $Disclosure\_policy_j^i$ is a vector of the disclosure policy for each observation. In the section above, we argued that early disclosure could make vendors patch the vulnerability more promptly. In addition to that, due to the different process of handling the vulnerability report between CERT/CC and SecurityFocus, there might be a potential selection bias from if the vulnerability is published by CERT/CC, i.e. the vulnerability published by CERT/CC may be systematically different from the vulnerability published by SecurityFocus in certain way that may affect the vendor patching speed. We call it CERT effect in this paper. $Disclosure\_policy_j^i$ measures early disclosure, CERT/CC disclosure and the interaction between this two. B_1 captures the effect of disclosure policy on patching time. $vendor\_characteristics^i$ is a set of vendor characteristics for vendor *i*. B_2 is a vector that measures the effect of vendor characteristics on the patching time. $Vul\_characteristics_j$ is a set of vulnerability characteristics for vulnerability *j*. B_3 is a vector that measures the effect of vulnerability characteristics on the patching time. Finally, ε_j^i capture the effect of unspecified variables.

Table 5-1 provides bivariate comparisons of the effects of disclosure source and timing on average patching time. To avoid the result being driven by few extreme values, we truncate the patching time to 365 days for the observations with patching time greater than 365 days in this comparison.

The first panel in table 5-1 compares the average patching time with early disclosure and without disclosure, which shows that the vulnerabilities that are disclosed early (disclosed within 10 days of notification) are patched 20 days faster on average. The early disclosure shows strong impact on the vendor patching speed and the result is statistically significant.

The second panel in table 5-1 compares the average patching speed among the vulnerabilities published by CERT/CC alone and the vulnerabilities published by SecurityFocus alone. The result suggests that vulnerabilities disclosed by CERT/CC are patched 17 days faster on average. And the t test result is about significant. Recall our discussion in the previous section that SecurityFocus tends to disclose earlier than CERT/CC. All else equal, the early disclosure should lead to a shorter patching time for the vulnerabilities published by SecurityFocus than by CERT/CC. Therefore, there are differentiating factors between CERT/CC and SecurityFocus, other than disclosure policy, which affect the vendor patching speed.

One of these factors could be disclosure processing and disclosure quality. SecurityFocus publishes all the raw vulnerability information from the discussion on the Bugtrag mailing list without sufficient control on the information quality. On the other hand CERT/CC researches vulnerabilities and chooses the most sever ones to work on. Moreover, CERT/CC also directly communicates with the appropriate employees of the vendors and coordinates with vendors on the patch development. We believe that these efforts may reduce patch development cost, and hence, patching times.

Panel three tests whether the source of early disclosure matters. It shows that the early disclosure by CERT/CC leads to an advantage of 12 days than the early disclosure by SecurityFocus and the result is statistically significant. This result indicates that the source of early disclosure does have impact on the vendor's patching speed.

Result set (4) suggests that notification matters. The vendors with pre-notification patch average 9 days faster than vendors without pre-notification. And the mean patching time difference is statistically significant. Therefore the communication with vendor on the vulnerability could make big difference. There are 73.7% of the observations published by CERT/CC that vendors are pre-notified. While only 46.5% of the observations published by SecurityFocus that vendors are pre-notified. This shows very strong lines of communication between CERT/CC and vendors compare to SecurityFocus.

Table 5-1 Bivariate comparisons of effects of disclosure source and timing on average patching time

Patching time		N	Mean	Std. Dev.	Min	Max
(1) Average effect of Early disclosure						
Early disclosure		883	43.51	78.86	1	365
Without early disclosure		397	63.17	79.70	1	365
Ha: Mean(diff)<0	*t= -4.0959*	*P<t = 0.00002*				
(2) Average effect of CERT/CC publication						
Disclosed by CERT/CC alone		80	46.99	84.61	1	365
Disclosed by SecurityFocus alone		99	63.91	94.79	1	365
Ha: Mean(diff)<0	*t=-1.26039*	*P<t = 0.103764*				
(3)Impact of the source of early disclosure on patching time (among the vulnerability jointly published by both)						
Early disclosed by CERT/CC		211	32.98	63.11	1	365
Early disclosed by SecurityFocus		526	45.17	82.09	1	365
Ha: Mean(diff)<0	*t=-2.16519*	*P<t =0.015187*				
(4 Average effect of notification						
With notification		916	47.13	75.25	1	365
Without notification		364	55.84	89.46	1	365
Ha: Mean(diff)<0	*t=-1.6418*	*P<t =0.050319*				

Table 5-2 shows the result of log linear regression. The vulnerability specified effect is controlled by vulnerability severity metric. We exclude the vulnerability technical dummy variables in the regression to avoid multi-collinearity since we've been using them to create vulnerability metric value for the SecurityFocus vulnerability. Column (1) has vendor fixed effects, while column 2 shows results where vendor characteristics such as size and firm type are used.

Specification (1.2) and (2.2) focuses on the effect of early disclosure without control for the source of early disclosure. Early disclosure regardless of the source shows strong positive effect on the patching speed (the coefficient is negative and statistically significant). The estimated results suggest that the mean patching time of the vulnerability early disclosed is only about 31% to 35% (calculated by taking exponential of the estimated coefficient) of the patching time of the vulnerability that are not early disclosed. Moreover, with specification (2.2), which includes the vendor characteristics, CERT/CC turns out to be significant. The estimated coefficient -0.44 shows that the mean patching time of the vulnerabilities published by CERT/CC is about 64% of the vulnerability that are not published by CERT/CC.

We take into consideration of the source of the early disclosure in specification (1.1) and (2.1). Our results show that whether the source of the early disclosure is CERT/CC or someone else make difference to the vendor patching speed. According to our result, with early disclosure by CERT/CC, the mean patching time is reduced to 40%. While with early disclosure by SecurityFocus, the mean patching time is reduced to 50%. Note that the control group in this case is the observations disclosed only on SecurityFocus and after 10 days of notification.

The communication with vendor always helps. The variable Notification in our result is estimated to be significant under all the specification with a coefficient of -0.45 to -0.57, which

shows that mean patching time could be reduced up to 60% if the vendors are explicitly contacted regarding to the vulnerability.

With the specification of (2.1) and (2.2), where the vendor characteristics are used for control, we observe that estimation of open source is statistically significant. The estimated coefficient indicates that open source software takes on average about 60% of the patching time than closed source software would take. With control for the vendor fixed effect, vulnerability severity becomes significant. One standard deviation (17.59) increase of the severity metric is estimated to reduce the patching time by about 10%. Finally, our result also shows that after September 11, vendor patching time is about 30% than before.

Table 5-2 Log linear regression: Dependent Variable =Log (Patching Time)

	With vendor fixed effect (1)		Vendor characteristics (2)	
	(1.1) interaction	(1.2) No interaction	(2.1) interaction	(2.2) No interaction
Constant	2.44 (2.00)	3.72* (1.96)	4.28*** (0.19)	4.91*** (0.24)
Early		-1.06*** (0.13)		-1.16*** (0.11)
CERT	1.71 (2.17)	0.83 (2.15)	0.14 (0.19)	-0.44*** (0.17)
Early_Comm_C	-0.97*** (0.17)		-1.00*** (0.14)	
Early_Comm_S	-0.71*** (0.12)		-0.91*** (0.11)	
Notify	-0.48*** (0.13)	-0.56*** (0.13)	-0.45*** (0.11)	-0.58*** (0.11)
Small and foreign Vendors			0.06 (0.14)	-0.07 (0.14)
Employee size			-7.69e-7 (6.72e-7)	-7.66E-07 (-6.64E-07)
Public firm			0.05 (0.13)	0.06 (0.13)
Open source software			-0.53*** (0.14)	-0.49*** (0.14)
Vulnerability severity metric	-0.006** (0.003)	-0.007*** (0.002)	0.0004 (0.002)	0.002 (0.002)
Post September/11	-0.37*** (0.12)	-0.50*** (0.12)	-0.32*** (0.10)	-0.45*** (0.10)

Notes: * indicates significant at 10% level, ** indicates significant at 5% level and *** indicates significant at 1% level.

5. Conclusion

Our paper provides critical empirical estimates on vendor response. To our knowledge, this is the first systematic empirical examination of this question, and marks a contribution towards understanding how vulnerability information should be disclosed.

Our results clearly show that early disclosure could make vendors patch faster. This verifies the previous literature on disclosure policy that vendors respond faster to early disclosure since they need to compensate the addition loses that early disclosure may causes to them. We also find that vendors respond slower to vulnerabilities not disclosed by CERT/CC. This might reflect unmeasured differences in the severity and importance of vulnerabilities. It might also reflect the stronger lines of communication between CERT/CC and vendors, and the value of the vulnerability analysis by CERT/CC.

Open source vendors are more likely to patch when vulnerability is found in their products. Given the attention on the open source vs. closed source debate, this is an important finding since the quality of a software product also depends on the ex-post support a vendor provides (Arora, Caulkins and Telang 2003). Severe vulnerability takes less time to patch. Our results are therefore consistent with a rational model in which vendors internalize some of the customers' losses. Our result also shows that vendors are more responsible after September 11.

While our results are interesting, there are qualifications. Particularly, our model doesn't control for the quality of the patch. Additional resource been allocated to patch development could result in higher quality patch than a patch that is released sooner. Given the importance of these issues and little empirical work, we hope that our study paves the way for more research with new and possibly better data sources.

REFERENCES

Arbaugh, W. A., W. L. Fithen, and J. McHugh, "Windows of vulnerability: A case study analysis," IEEE Computer, vol. 33 (December 2000), pp. 52–59

Arora Ashish, Jonathan P. Caulkins, Rahul Telang, "Sell First Fix Later: Impact of Patching on Software Quality," Carnegie Mellon University, working paper, Jan. 2003

Arora Ashish, Rahul Telang and Hao Xu, "Timing Disclosure of Software Vulnerability for Optimal Social Welfare," Carnegie Mellon University working paper, April 2004

Breslow. N. E., "Covariance analysis of censored survival data", Biomerics, 1974, 30, 89-99.

Cox, David R. "Regression Models and Life-Tables (with discussion)," J. Roy. Statist. Soc., May/Aug. 1972, B 34, PP. 187-220

Cox, David R., "Partial likelihood", Biometrika, May/Aug., 1975, 62(2), pp. 269-76.

Elias, L. Full Disclosure is a necessary Evil, SecurityFocus.com, www.securityfocus.com/news/238, 2001

Farrow, R., The Pros and Cons of Posting Vulnerability, The Network Magazine, www.networkmagazine.com/shared/article, 2000

Gordon. S and Richard Ford, "When Worlds Collide: Information Sharing for the Security and Anti-virus Communities," IBM research paper, 1999

Kalbfleisch, J. D., and Prentice, R. L. The Statistical Analysis of Failure Time Data second edition. John Wiley and Sons, 2002

Lawrence A. Gordon, Martin P. Loeb and Tashfeen Sohail, "A framework for using insurance for Cyber-Risk Management" Communications of the ACM Volume 46(March, 2003), pp. 81 – 85

Meyer, Bruce. 1990. "Unemployment Insurance and Unemployment Spells." Econometrica 58(4): 757-83

Nicholas M. Kiefer, "Economic Duration Data and Hazard Functions" Journal of Economic Literature, Vol.26, No.2 (Jun., 1988), pp. 646-679

Peto, R., "contribution to the discussion of paper by D.R. Cox. J. Roy", Statist. Soc., 1972, Sec. B, 34, 205-207.

Steve Beattie, S. A., Crispin Cowan, Perry Wagle, and Chris Wright, Timing the Application of Security Patches for Optimal Uptime. Proceedings of LISA 2002: 16th Systems Administration Conference, Philadelphia, PA, 2002

Varian H R, "Managing Online Security Risks," The New York Times, http://www.nyt imes.com/library/financial/columns/060100econ-scene.html, 2000

Vaupel, J. W., Manton, K.G., and Stallard, E., The impact of heterogeneity in individual frailty on the dynamics of mortality, Demography, (1979), 16, 439-454