6-2023

Enhancing third-party software reliability through bug bounty programs

Tianlu ZHOU

Dan MA
Singapore Management University, madan@smu.edu.sg

Nan FENG

Citation

ZHOU, Tianlu; Dan MA; and FENG, Nan. Enhancing third-party software reliability through bug bounty programs. (2023). *Proceedings of the16th China Summer Workshop on Information Management, Changsha, China, 2023 June 24-25*. 169-174.
Available at: https://ink.library.smu.edu.sg/sis_research/8596

Enhancing Third-Party Software Reliability Through Bug Bounty Programs

Tianlu Zhou
Tianjin University
imis_ztl@tju.edu.cn

Dan Ma
Singapore Management University
madan@smu.edu.sg

Nan Feng
Tianjin University/Tianjin University (Qingdao) Ocean Engineering Research Institute Co.
fengnan@tju.edu.cn

Abstract

Bug Bounty Programs (BBPs) reward external hackers for identifying and reporting software vulnerabilities. As the number of security issues caused by third-party applications has been significantly increased recently, many digital platforms are considering launching BBPs to help enhance the reliability of third-party software. BBPs bring benefits to the platform and vendors, meanwhile impose additional costs on them as well. As a result, the overall impact of using BBP is unclear. In this paper, we present an analytical model to examine the strategic decisions of launching and participating in a BBP for the platform and the third-party vendor, respectively. We find that the platform's (the vendor's) BBP launching (participation) decisions depend on two key factors: the expected loss due to security breaches and the vendor's initial reliability investment efficiency. We show that the incentive of using BBP, for the platform and vendor, sometimes is inconsistent.

Keywords: bug bounty program, digital platform, third-party application

1. Introduction

Nowadays, third-party software plays a critical role on digital platforms. According to Gartner, cybercriminals are increasingly using third parties to attack crucial targets. Digital platforms, therefore, are actively looking for ways to enhance the reliability of third-party software to mitigate security breach loss. Many platforms have launched Bug Bounty Programs (BBPs): Platforms offer financial rewards to external ethical hackers for testing third-party software and reporting valid vulnerabilities. Facebook, for example, pays hackers for reporting security bugs in third-party software. Platforms are expecting several benefits of using BBPs. First, under BBPs, external hackers are motivated to legally report bugs and vulnerabilities, instead of using them to launch attacks maliciously, leading to threat reduction (Zhou & Hui, 2021). In addition, with BBPs, software vendors will verify and fix valid bugs reported by external hackers and thus further improve the reliability of their software, which makes the platform a more secure marketplace and thus obtain higher profits.

Not all platforms, however, choose to launch BBPs for third-parties. Apple is one example. It decides not to reward hackers for reporting bugs in the third-party software. One obvious reason is the costs associated with BBPs that the platform must pay bug bounty rewards. Besides, the platform also needs to take the third-part vendors' reaction into considerations. Knowing there will be ethical hackers who help to discover vulnerability of their products in the later stage, will the third-party vendor choose to reduce the initial investment in the software reliability investment? If so, what will be the ultimate software reliability level? This is an unclear issue. Thus, the platform must find out under what conditions it is profitable to launch a BPP for the third-party software and understand how its launching decision impact would the vendor's behavior. This is the first research question we will address in this work.

Only being authorized and verified by the third-party vendor, external hackers can earn the rewards provided by the platform. The third-party vendor, however, could choose not to participate in BBPs launched by the platform if it finds not worthwhile to do so. It must evaluate and tradeoff the expected benefits and costs. On the positive side, if the vendor participates in the BBP, it enjoys the threat reduction and potential revenue increase as the platform does. Meanwhile, the vendor incurs BBP related costs of processing bug reports and fixing valid bugs raised by external hackers. In addition, with the help of external hackers, the vendor shall adjust its software reliability investment accordingly. Hence, the second research question of our work is related to the software vendor's optimal decisions: When should the third-party vendor participate in a BBP launched by the platform? If so, what is the vendor's optimal level of initial software reliability investment?

2. Literature Review

Our work relates to three streams of research. The first one is the BBP literature. Recent studies on BBPs mostly focus on its performance (e.g., Zhou & Hui, 2021). Different from these works, ours is the first one to examine the optimal BBP adoption decisions for both the platform and the third-party software vendor. The second relevant research stream is about how various software testing methodologies can enhance software reliability. Earlier works in this stream have mainly focused on examining the effects of in-house testing on software reliability (e.g., Ji et al., 2005). Unlike them, we analyze the external ethical hackers' contribution to the improvement of software reliability, together with the third-party vendor's software reliability investment. Finally, the literature that studies how digital platforms manage third-party software to maximize profit is also relevant to us. Previous research has primarily focused on whether platforms should be corporate with third parties (Boudreau, 2010; Parker et al., 2017), and how to provide integration tools such as APIs and SDKs to third-party vendors (Tan et al., 2020; Xue et al., 2019). Our paper differs from the existing literature in that we investigate when the platform shall adopt the BBP to prevent security incidents caused by third-party software.

3. Model

Consider a digital platform (PF) on which the third-party vendor (VD) develops, lists, and sells its own software applications. Throughout this paper, we denote L and NL as the platform's strategy of launching and not-launching BBP, and P and NP as the vendor's strategy of participating and not-participating BBP (in the case that BBP is launched by the platform). We then have two types of outcomes: BBP (i.e., (L, P)), and NBBP (i.e., (L, NP), (NL, P), or (NL, NP)). We use BBP and NBBP as lower subscripts to indicate the two different outcomes.

External Hackers: We normalize the total number of external hackers to be 1, and assume they are all malicious when there is no BBP. When there is BBP, some hackers will convert to be ethical. They report discovered bugs to the platform for monetary rewards, instead of launching an attack. The platform will pay the reward r to the ethical hacker for each valid bug (i.e., after being verified by the vendor). It is intuitive that when this reward amount r is higher, more will turn to be ethical hackers. Following previous empirical studies (Zhao et al., 2015), we assume the number of hackers who convert from malicious to ethical as αr, where α is hackers' reward sensitivity coefficient, $0 < \alpha r < 1$. Denote n_e and n_m as the number of ethical and malicious hackers respectively. In the NBBP outcome, $n_e = 0$ and $n_m = 1$; and in the BBP outcome, $n_e = \alpha r$, $n_m = 1 - \alpha r$.

The Vendor (VD): The reliability level of third-party software is denoted as S. Let 1 be the highest possible reliability level, $S < 1$. This assumption reflects the fact that there is no completely bug-free software in practice. Further, we define the vulnerability of the third-party

software as $p = 1 - S$. Here, p also could be interpreted as the probability that external hackers discover valid bugs in the software.

The reliability of software, S, consists of three components. First, the software has an intrinsic reliability level S_0. It depends on the quality of the software itself, lies in the interior design of the application, and varies in terms of software types, programming languages and skills.

Second, to further improve reliability, the software vendor usually will make additional investments (such as ongoing software testing), which is referred as *vendor's reliability investment* in this study. We denote the vendor's reliability investment efforts as βz^2, where z is the resulted reliability improvement level, β is the investment cost coefficient that indicates the vendor's efficiency level in reliability investment, and the quadratic form represents the diminishing investment return. In the case of NBBP - either the platform does not launch a BBP or the vendor chooses not to participate, the software reliability level is the sum of these two components: $S_{NBBP} = S_0 + z_{NBBP}$, where the lower subscript NBBP indicates it is under the case without BBP.

In the case of BBP - ethical hackers will discover and report bugs to the platform to gain rewards. After the vendor verifies valid bugs and fixes them, the software reliability level could be further improved. We recognize that it might take a while for the bugs to be identified and fixed, but this period should be transient. Therefore, we use upper superscripts B and A to indicate the software reliability level before and after such a transient period. As a result, the reliability level before the transient period (i.e., before bugs are reported and fixed) can be written as $S_{BBP}^B = S_0 + z_{BBP}$, where the lower subscript BBP indicates it is under the case with BBP. During the transient period, there are $p_{BBP}^B n_e$ valid bugs to be fixed by the vendor, where $p_{BBP}^B = 1 - S_{BBP}^B$ is the probability that a hacker identifies a valid bug and $n_e = \alpha r$ is the total number of ethical hackers under the BBP reward amount r. Hence, after the transient period (i.e., after bugs have been fixed), the software reliability is improved and reaches to $S_{BBP}^A = S_{BBP}^B + p_{BBP}^B n_e f$, where f is the reliability level improvement due to the fix of each valid bug. We can also view f as the efficiency of BBP since it represents the unit reliability level improvement due to the BBP use. Consequently, the vulnerability of the third-party software reduces to $p_{BBP}^A = 1 - S_{BBP}^A$.

Without loss of generality, we assume that the expected revenue generated from the third-party software increases in software reliability, SR, where R is the highest possible revenue when the software is completely free of bugs (i.e., when $S = 1$). The expected revenue is shared by both the vendor and platform: the vendor obtains θ percent and the platform $1 - \theta$ percent. In addition, if a malicious hacker launches an attack and results in security breach, the software vendor and platform incur utility losses λ_{VD} and λ_{PF} respectively.

When BBP is offered by the platform, the vendor has two strategies to consider:

Not participate (NP): If the vendor chooses not to participate, then the outcome is NBBP. The vendor derives expected revenues $\theta S_{NBBP} R$, pays the reliability investment cost βz_{NBBP}^2, and incurs the expected loss $p_{NBBT} n_m \lambda_{VD}$. The total payoff of the vendor is given by

$$\Pi_{NP}^{VD} = \theta S_{NBBP} R - (\beta z_{NBBP}^2 + p_{NBBT} n_m \lambda_{VD}). \quad (1)$$

Participate (P): The vendor derives expected revenues $\theta S_{BBP}^A R$ from the third-party software, pays the reliability investment cost βz_{BBP}^2, and incurs potential loss $p_{BBP}^A n_m \lambda_{VD}$. Let c_p and c_f represent the vendor's unit processing and fixing cost, respectively. Thus, the vendor bears the total processing costs $n_e c_p$, as well as the total fixing costs $p_{BBP}^B n_e c_f$. The total payoff of the software vendor is given by

$$\Pi_P^{VD} = \theta S_{BBP}^A R - (\beta z_{BBP}^2 + p_{BBP}^A n_m \lambda_{VD} + n_e c_p + p_{BBP}^B n_e c_f). \quad (2)$$

The Platform (PF): Similarly, the platform can choose one of two options:

Not Launch (NL): The platform derives expected revenues $(1-\theta)S_{NBBP}R$ from the third-party software and incurs the expected loss $p_{NBBP}n_m\lambda_{PL}$ caused by potential software breaches. The total payoff of the platform is

$$\Pi_{NL}^{PF} = (1-\theta)S_{NBBP}R - p_{NBBP}n_m\lambda_{PF}. \quad (3)$$

Launch (L): If the vendor chooses NP, the platform's payoff is the same as the above NL case, given in Equation (3). If the vendor chooses P, the platform derives expected revenues $(1-\theta)S_{BBP}^{A}R$ from the third-party software, incurs the expected loss $p_{BBP}^{A}n_m\lambda_{PL}$, and pays hackers who reported valid bugs the BBP rewards $p_{BBP}^{B}n_e r$. As a result, the total payoff of the platform can be written as

$$\Pi_{L}^{PF} = (1-\theta)S_{BBP}^{A}R - (p_{BBP}^{A}n_m\lambda_{PF} + p_{BBP}^{B}n_e r). \quad (4)$$

4. Analysis and Results

After the software vendor decides to whether to participate the BBP program, it should determine the optimal investment level z_i in software reliability accordingly, where $i = BBP\ or\ NBBP$. We have the following findings.

Proposition 1. **(Optimal Reliability Investment Level)**

a. *If the vendor participates in the BBP launched by the platform, the optimal reliability investment level is* $z_{BBP}^{*} = \frac{c_f r\alpha + (1-fr\alpha)(R\theta+\lambda_{VD}(1-r\alpha))}{2\beta}$*; if it does not participate, the optimal reliability investment level is* $z_{NBBP}^{*} = \frac{R\theta+\lambda_V}{2\beta}$.

b. *If the BBP-related bug-fixing cost is low,* $c_f < \lambda_{VD} + f(R\theta + \lambda_{VD}(1-r\alpha))$, $z_{BBP}^{*} < z_{NBBP}^{*}$*; otherwise,* $z_{BBP}^{*} \geq z_{NBBP}^{*}$.

Proposition 1 shows that when the third-party vendor chooses to participate in the BBP, its initial software reliability investment increases with the bug-fixing cost. It is because both the BBP and the vendor's initial investment serve to enhance the software reliability and thus help reduce the potential breach loss. If the BBP is "expensive," i.e., incurring in relatively high bug-fixing costs, the vendor will instead increase its initial reliability investment. When the bug-fixing cost continues to increase and reach a certain threshold c_f, the vendor's optimal reliability investment level under participation becomes even higher than that under non-participation.

Proposition 2. (Software Vendor's Participation Decision)

The third-party software vendor should participate in the BBP if and only if: (1)its potential loss due to security breaches is high, $\lambda_{VD} > \lambda_{VD1}$ *AND (2) its reliability investment efficiency is low,* $\beta > max\{\beta_{VD}, \beta_1\}$.[1]

Proposition 2 reveals the fundamental economic rationale of the third-party software vendor's BBP participation decision. Intuitively, when the potential loss of security breach is significant, the vendor needs to actively search ways to improve the software reliability. For the vendor, there are two viable ways: increasing its initial reliability investment or leveraging hacker community through participating BBP. If the vendor's reliability investment efficiency is high, it will opt for the first way only (i.e., not participating in the BBP) because its reliability investment alone can reduce potential loss sufficiently, which also avoids the BBP-related costs. But if the vendor's reliability investment is inefficient, the BBP needs to adopt the BBP as a way to complement its own reliability investment.

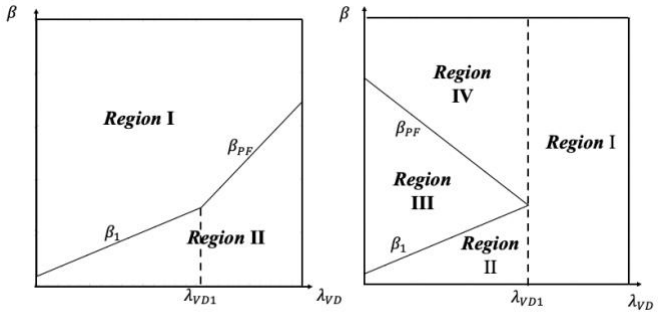
We now analyze when it is profitable for the platform to launch the BBP for the third-party vendor.

[1] We define $\lambda_{VD1} = \frac{c_f - fR\theta}{1+f-fr\alpha}$, $\beta_1 = \frac{R\theta - frR\alpha\theta + rac_f + \lambda_{VD}(1-r\alpha-fr\alpha+fr^2\alpha^2)}{2-2q}$ and $\beta_{VD} = \frac{(c_f-\lambda_{VD}-f(R\theta+\lambda_{VD}-r\alpha\lambda_{VD}))(2(R\theta+\lambda_{VD})+r\alpha(c_f-\lambda_{VD}-f(R\theta+\lambda_{VD}-r\alpha\lambda_{VD})))}{4(1-q)c_f+4(c_p-(1-q)(\lambda_{VD}+f(R\theta+\lambda_{VD}-r\alpha\lambda_{VD})))}$.

Proposition 3. (Platform's Launch Decision)
The platform will choose to launch the BBP:

a. *when the platform's potential loss is high,* $\lambda_{PF} \geq \lambda_{PF1}$*, and the vendor's reliability investment efficiency is low,* $\beta > \beta'$ *where* $\beta' = \begin{cases} \beta_1 & if\ \lambda_{VD} < \lambda_{VD1} \\ \beta_{PF} & if\ \lambda_{VD} \geq \lambda_{VD1} \end{cases}$.

b. *when the platform's potential loss is low,* $\lambda_{PF} < \lambda_{PF1}$*, and the vendor's potential loss is low* $\lambda_{VD} < \lambda_{VD1}$ *and reliability investment efficiency is moderate* $\beta \in [\beta_1, \beta_{PF}]$.[2]



(a) $\lambda_{PF} \geq \lambda_{PF1}$ (b) $\lambda_{PF} < \lambda_{PF1}$

Figure 1. The Platform's Launch Strategies of the BBP

Described by Proposition 3a and the region I in Figure 1(a), intuitively, the platform is more willing to launch BBP if it expects a high potential breach loss and meanwhile when the third-party vendor is not efficient in making software reliability investment. The other case when the platform should opt for launching BBP is described by Proposition 3b and demonstrated in region III in Figure 1(b). Note that in this case, the platform expects a small loss even if a breach happens. Then what motivates the platform to launch BBP, given there are additional costs of doing so? This is a less intuitive but more interesting case. To understand the platform's choices in $\lambda_{PF} < \lambda_{PF1}$, we must take the vendor's reaction into considerations as well. Note that z^*_{BBP} increases in λ_{VD}, the vendor's potential loss due to security breaches. Hence, when $\lambda_{VD} > \lambda_{VD1}$, the vendor's initial reliability investment is sufficiently high so that the platform will not have interest in launching BBP to further improving software reliability, which is the region I of Figure 1(b).

Next, we analyze the case of $\lambda_{VD} \leq \lambda_{VD1}$. Note that z^*_{BBP} decreases in β, vendor's reliability investment efficiency. λ_{VD} and β thus have opposite effects on the optimal reliability investment level z^*_{BBP}. As a result, when both λ_{VD} and β are small, namely, $\lambda_{VD} < \lambda_{VD1}$ and $\beta < \beta_1$, as shown in the region II of Figure 1(b), vendor chooses to invest in software reliability at a moderate level. For the platform, the benefits of launching BBP are from reducing the potential loss caused by the security breaches. Since the platform's loss is low when a breach happens, it is likely that the platform would consider the vendor's initial reliability investment as sufficient. Thus, the platform should not launch the BBP in this region.

When β increases to a moderate level, $\beta \in [\beta_1, \beta_{PF}]$, the optimal reliability investment of the vendor decreases accordingly and becomes insufficient for the platform in the sense that the low software reliability induces too many software vulnerabilities and meanwhile generates too little revenue. Therefore, the platform now is willing to invest in the BBP to increase software reliability, expecting that it will obtain more revenues from end users and thus become more profitable. This is what happens in the region III in Figure 1(b).

---

[2] We define $\beta_{PF} = \frac{1}{2(1-q)(fR(1-\theta)-r+(1+f-fr\alpha)\lambda_{PF})}(2R^2(1-\theta)\theta + R(r^2\alpha\theta - 2\beta(1-\theta)(1-q+r\alpha c_f) + 2(1-r\alpha)(\theta\lambda_{PF} + (1-\theta)\lambda_{VD})) + (1-r\alpha)(r^2\alpha\lambda_{VD} - \lambda_{PF}(2(1-q)\beta - r\alpha c_f - (2-2r\alpha)\lambda_{VD})) - \sqrt{(4r\alpha(R(\theta-1)-(1-r\alpha)\lambda_{PF})(R\theta+(1-r\alpha)\lambda_{VD})((2(q-1)\beta+R\theta)(r-\lambda_{PF}) + c_f(R(1-\theta)+r\alpha(r-\lambda_{PF})+\lambda_{PF}) + (r-r^2\alpha-R(1-\theta)-(2-r\alpha)\lambda_{PF})\lambda_{VD}) + (R(r^2\alpha\theta + 2R(1-\theta)(\theta-2\beta(1-q))) + r\alpha c_f(R(1-\theta)+(1-r\alpha)\lambda_{PF}) + (1-r\alpha)((r^2\alpha+2R(1-\theta))\lambda_{VD} - 2\lambda_{PF}((1-q)\beta - R\theta - (1-r\alpha)\lambda_{VD})))^2))$.

Finally, when β becomes very large, $\beta > \beta_{PF}$, the vendor will choose a very low reliability level in the first place. If the platform launches the BBP under such a situation, it must pay significant rewards for many reported bugs because of the poor initial software reliability. The cost of launching a BBP exceeds the revenue gains and breach loss savings. So the platform again loses interest in launching BBP, as shown in region IV in Figure 1(b).

Proposition 4. (Equilibrium Outcome)

When the potential loss of both the vendor and platform are high, $\lambda_{VD} \geq \lambda_{VD1}$ and $\lambda_{PF} \geq \lambda_{PF1}$, and when the vendor's reliability investment efficiency is low, $\beta > \beta_{BBP} = max\{\beta_{VD}, \beta_{PF}\}$, the equilibrium outcome is BBP; Otherwise, it is NBBP.

The BBP equilibrium appears only when security breach will cause significant loss for both the vendor and platform and when the vendor cannot efficiently invest in software reliability alone. As a result, both the platform and vendor are willing to pay extra efforts to leverage external hackers' capability to improve software security.

5. Conclusion

In this paper, we model the interactions between a platform and a third-party software vendor in considering using BBP to improve software reliability. Our findings show that not all vendors prefer to participate the BBP. Meanwhile, we find that that even when the potential loss of the vendor and the platform are relatively low, the platform may still have an incentive to launch a BBP for those vendors who have moderate reliability investment efficiency. However, vendors could refuse to participate in such a BBP. We conclude that the BBP is profitable for both the vendor and platform only when the vendor is inefficient in the software reliability investment and when the potential loss for both are high.

Acknowledgements

The authors wish to acknowledge the National Natural Science Foundation of China (72231004, 71871155), China Scholarship Council (202206250105), and the Shandong Industrial Internet Innovation and Entrepreneurship Community for financial support.

REFERENCES

Boudreau, K. 2010. "Open platform strategies and innovation: Granting access vs. devolving control," *Management Science (56*:10), pp. 1849–1872.

Ji, Y., Mookerjee, V. S, and Sethi, S. P. 2005. "Optimal software development: A control theoretic approach," *Information Systems Research (16*:3), pp. 292-306.

Parker, G., Van, A. M., and Jiang, X. 2017. "Platform ecosystems: How developers invert the firm," *MIS Quarterly (*41:1), pp. 255–266.

Tan, B., Anderson, E.G., and Parker, G.G. 2020. "Platform pricing and investment to drive third-party value creation in two-sided networks," *Information Systems Research (*31:1), pp. 217–239.

Xue, L., Song, P., Rai, A., Zhang, C., and Zhao, X. 2019. "Implications of application programming interfaces for third-party new app development and copycatting," *Production and Operations Management, (*28:8*)*, pp. 1887–1902.

Zhao, M., Aron L., and Jens G. 2017. "Devising effective policies for bug-bounty platforms and security vulnerability discovery," *Journal of Information Policy (*7: 1), pp. 372-418.

Zhou, J., & Hui, K.L. 2021. "Sleeping with the enemy: An economic and security analysis of bug bounty programs," *Hong Kong University of Science and Technology.* https://ssrn.com/abstract=3940307.