Optimal Policy for Software Vulnerability Disclosure

Ashish Arora, Rahul Telang, Hao Xu
2008

Presented by Chongyang Shi
Feb 5th, 2018

Software Vulnerabilities

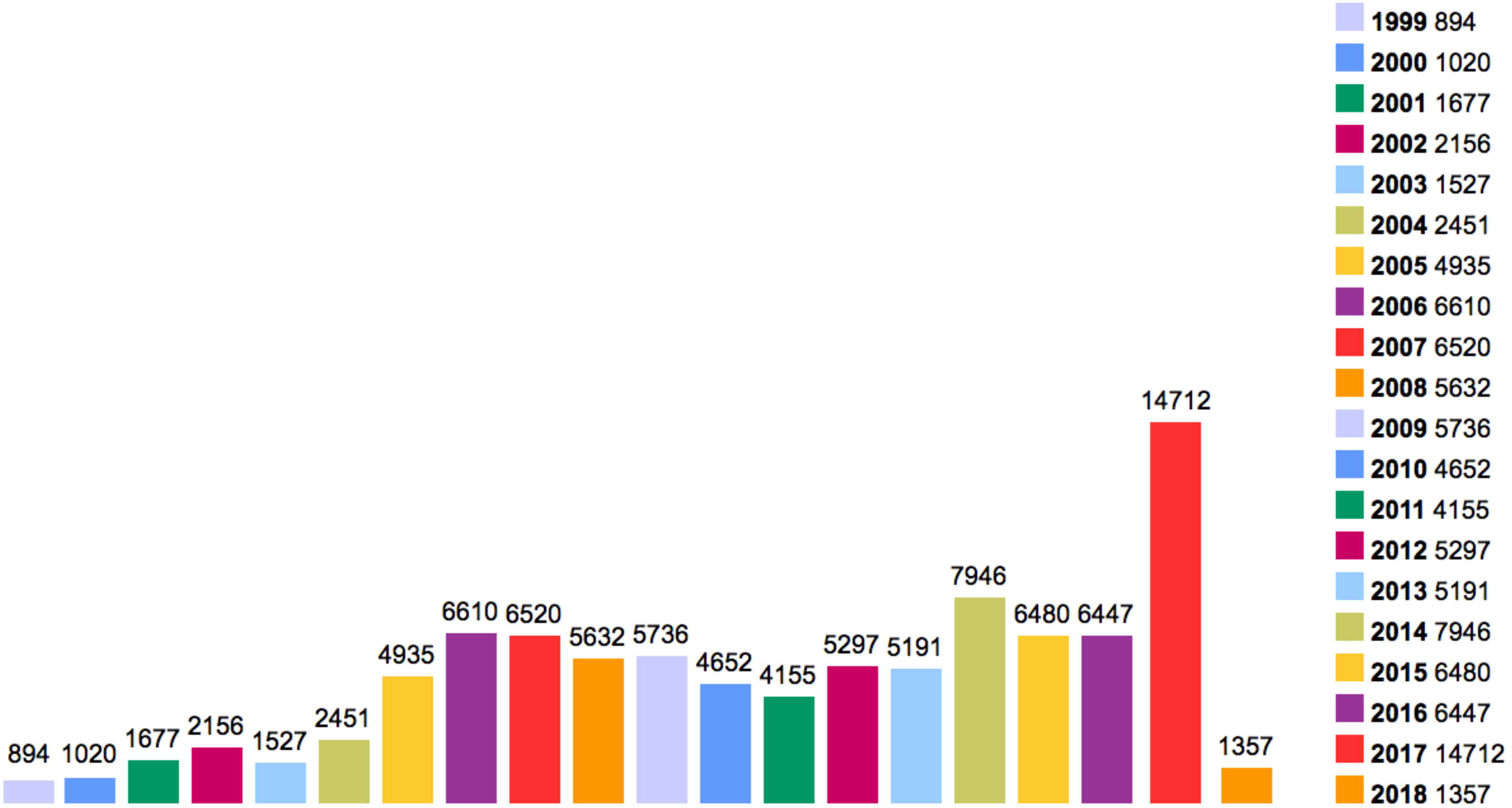
- “A security vulnerability is a flaw within a software product that can cause it to work contrary to its documented design and can be exploited to cause the system to violate its documented security policy.” (Telang and Wattal, 2007).
- Symantec report: 2,524 vulnerabilities in 2002, an 81.5% increase from 2001.
- CERT/CC: 8,064 vulnerabilities* in 2006.

* As reported by the authors, differing from the MITRE figure.

Software Vulnerabilities

- General increasing trend of reported vulnerabilities.

- 2017 spike partly due to significant increase of vulnerabilities in Android and Linux Kernel.



Reported CVEs by year (MITRE/CVE Details)

Software Vulnerabilities

- Software vulnerabilities incur costs to users and vendors alike.

- Estimated $10 billion spent by users in botnet clean-up (Anderson et al, 2013), with a significant proportion of infections due to vulnerabilities.

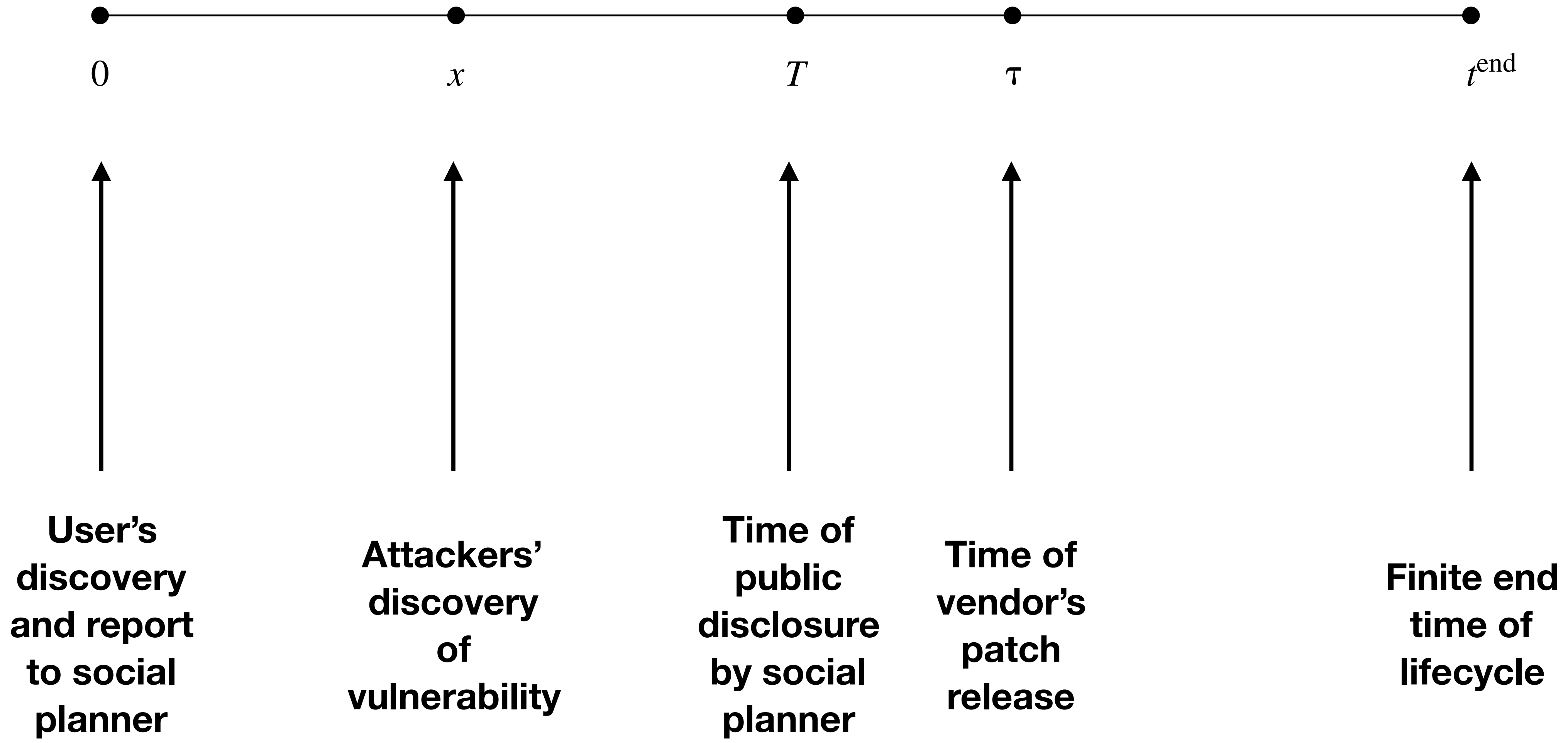
- Software vendors are not financially liable for their vulnerabilities in most jurisdictions, but vulnerabilities cost them through market effects and support contracts.

- Due to misaligned incentives (Anderson and Moore, 2009), vendors normally only internalise a proportion of customer loss, and can take a long period of time to patch vulnerabilities to minimise cost.

Vulnerability Disclosure

- How vulnerabilities should be disclosed is a subject of debate.
- Some users resort to full, instant disclosures without a patch, which could be dangerous to other users.
- Third party purchase and resale of vulnerabilities commonplace.
- This model: the role of a *social planner* to coordinate a protected period (e.g. 30 or 45 days) for the vendor to patch a vulnerability before public disclosure.
- This study: how the protected period can be best determined to minimise social cost under different circumstances.

Timeline of Vulnerability



0 User’s discovery and report to social planner

x Attackers’ discovery of vulnerability

T Time of public disclosure by social planner

τ Time of vendor’s patch release

t^{end} Finite end time of lifecycle

For which, the social planner chooses the optimal disclosure time T*, while the vendor chooses a patch development time τ*.

Use of the Model

- Optimally, the social planner will decide on its optimal disclosure time T* directly according the model, to which the vendor responds with an optimal development time τ*.

- However, this is never how it works in practice, so…

- The social planner will disclose the vulnerability to the vendor for it to set a static patch development time; and the planner then chooses a public disclosure time based on the model.

Assumptions: Pre-patch Loss

Assumption 1 (A1). *$l(y)$ is increasing and strictly convex in y, $t^{\text{end}} \geq y \geq 0$ and $l(0) = 0$.*

- The longer the users are exposed to an unpatched vulnerability, the greater cumulative loss they will suffer, due to increases in exploitation over time.

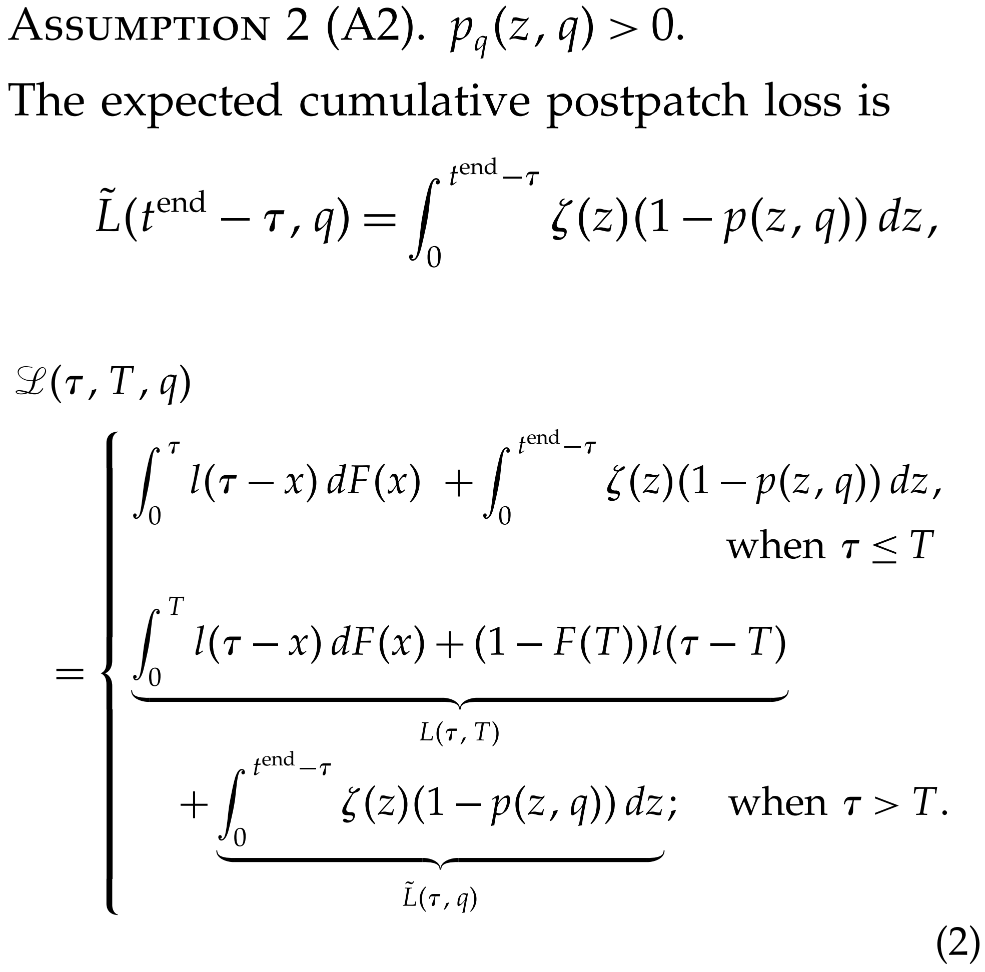
Patch released before disclosure but after attacker discovery.

$$L(\tau, T) = \begin{cases} \displaystyle\int_0^{\tau} l(\tau - x)\, dF(x), & \text{when } \tau < T \\ \displaystyle\int_0^{T} l(\tau - x)\, dF(x) + (1 - F(T))l(\tau - T), & \\ & \text{when } \tau \geq T. \end{cases} \tag{1}$$

Patch released after disclosure, attackers exploit during both periods.

Assumptions: Postpatch Loss

- Attackers can also exploit unpatched users after the release of the patch, due to not all users installing the patch immediately, subject to patch quality.



Assumption 2 (A2). $p_q(z, q) > 0$.

The expected cumulative postpatch loss is

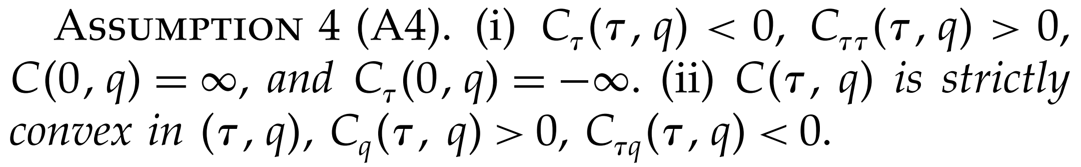
$$\tilde{L}(t^{\text{end}} - \tau, q) = \int_0^{t^{\text{end}} - \tau} \zeta(z)(1 - p(z, q))\, dz,$$

$$\mathscr{L}(\tau, T, q) = \begin{cases} \displaystyle\int_0^{\tau} l(\tau - x)\, dF(x) \ + \int_0^{t^{\text{end}} - \tau} \zeta(z)(1 - p(z, q))\, dz, & \\ \qquad\qquad\qquad\qquad\qquad\qquad \text{when } \tau \leq T & \\ \underbrace{\displaystyle\int_0^{T} l(\tau - x)\, dF(x) + (1 - F(T))l(\tau - T)}_{L(\tau, T)} & \\ \quad + \underbrace{\displaystyle\int_0^{t^{\text{end}} - \tau} \zeta(z)(1 - p(z, q))\, dz}_{\tilde{L}(\tau, q)}; \quad \text{when } \tau > T. & \end{cases} \tag{2}$$

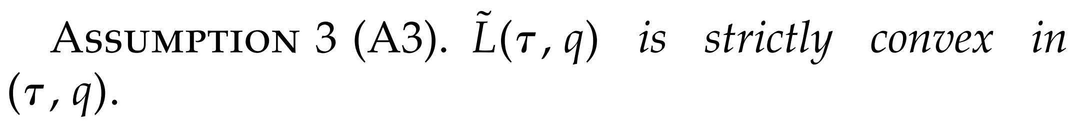
(Essentially, (1) plus instantaneous postpatch loss over time in either case.)

Assumptions: Patching

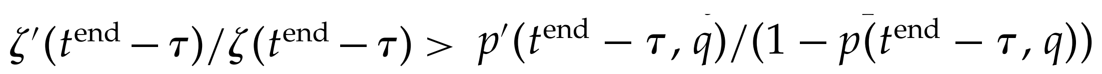
- The shorter time there is for the vendor to develop a patch, the more costly it is for the vendor.
- However, the decrease in cost diminishes over extended time.
- Users apply the patch at a slower rate than both the growth of postpatch loss and the increase in patch quality.



Assumption 4 (A4). (i) $C_{\tau}(\tau, q) < 0$, $C_{\tau\tau}(\tau, q) > 0$, $C(0, q) = \infty$, *and* $C_{\tau}(0, q) = -\infty$. (ii) $C(\tau, q)$ *is strictly convex in* (τ, q), $C_q(\tau, q) > 0$, $C_{\tau q}(\tau, q) < 0$.



Assumption 3 (A3). $\tilde{L}(\tau, q)$ *is strictly convex in* (τ, q).



$$\zeta'(t^{\text{end}} - \tau)/\zeta(t^{\text{end}} - \tau) > p'(t^{\text{end}} - \tau, q)/(1 - p(t^{\text{end}} - \tau, q))$$

Assumptions: The vendor

- We assume that it is always less costly for the vendor to patch a vulnerability before the end of the lifecycle rather than not patching at all.
- The vendor's expected cost function is a combination of development cost and its internalised user loss.

ASSUMPTION 5 (A5). $\min_\tau\{C(\tau,q)+\lambda\int_0^\tau l(\tau-x)\,dF(x)\}$
$+\lambda\int_0^{t^{\text{end}}-\tau}\zeta(z)\cdot(1-p(z,q))\,dz<\lambda\int_0^{t^{\text{end}}} l(t^{\text{end}}-x)\,dF(x).$

The vendor's expected cost function

$$V(\tau,q;T)=C(\tau,q)+\lambda\mathscr{L}(\tau,T,q).$$

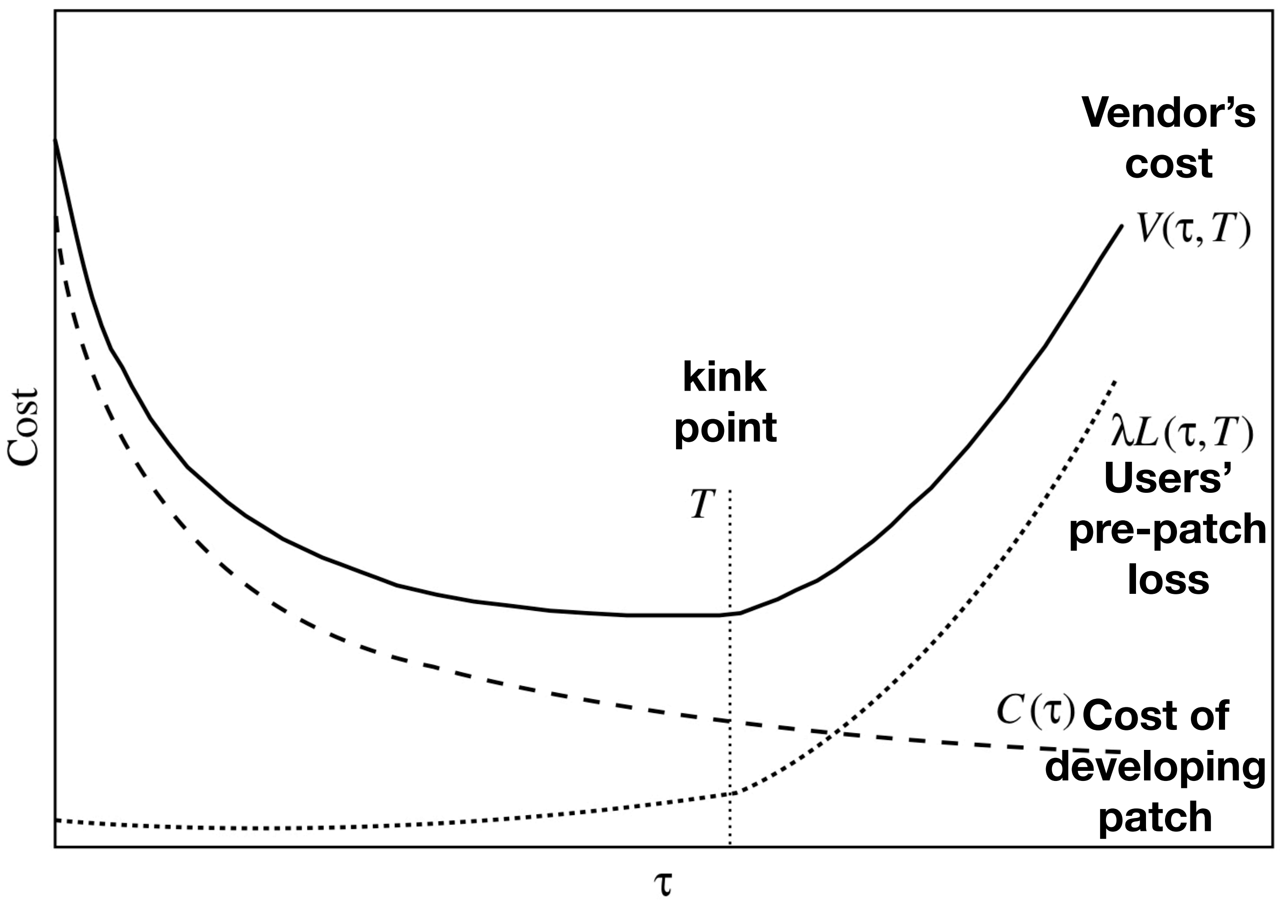
(for which at a set quality)

$$V(\tau;T)=C(\tau)+\lambda L(\tau,T). \qquad (3)$$

Optimal Decision: Kink Point

- Disclosures are often followed by a spike in attacks (Arora et al., 2006), even with a small gap between disclosure and patch release.

- The optimal patch release time τ* can be at τ* = T, where the vendor cost function is not differentiable, known as the kink point.



The vendor's expected cost function

$$V(\tau;\, T) = C(\tau) + \lambda L(\tau,\, T). \qquad (3)$$

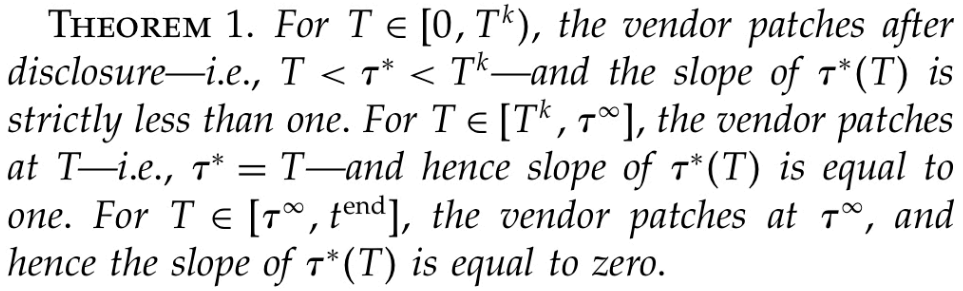
Optimal Decision: Vendor

- High-risk vulnerabilities will push forward the kink point and cause high loss should a patch be delayed, this increases the vendor's internalised user loss.

- To minimise the overall cost (development and internalised user loss), the vendor releases the patch at or after the kink point, depends on the time of public disclosure set by the social planner.

Socially optimal patching time

$$\tau^s = \arg\min_{\tau} \left\{ \underset{\substack{\big\uparrow \\ \textbf{Vendor} \\ \textbf{Cost}}}{C(\tau)} + \int_0^{\tau} \underset{\substack{\big\uparrow \\ \textbf{User} \\ \textbf{Loss}}}{l(\tau - x)}\, dF(x) \right\}. \qquad (4)$$



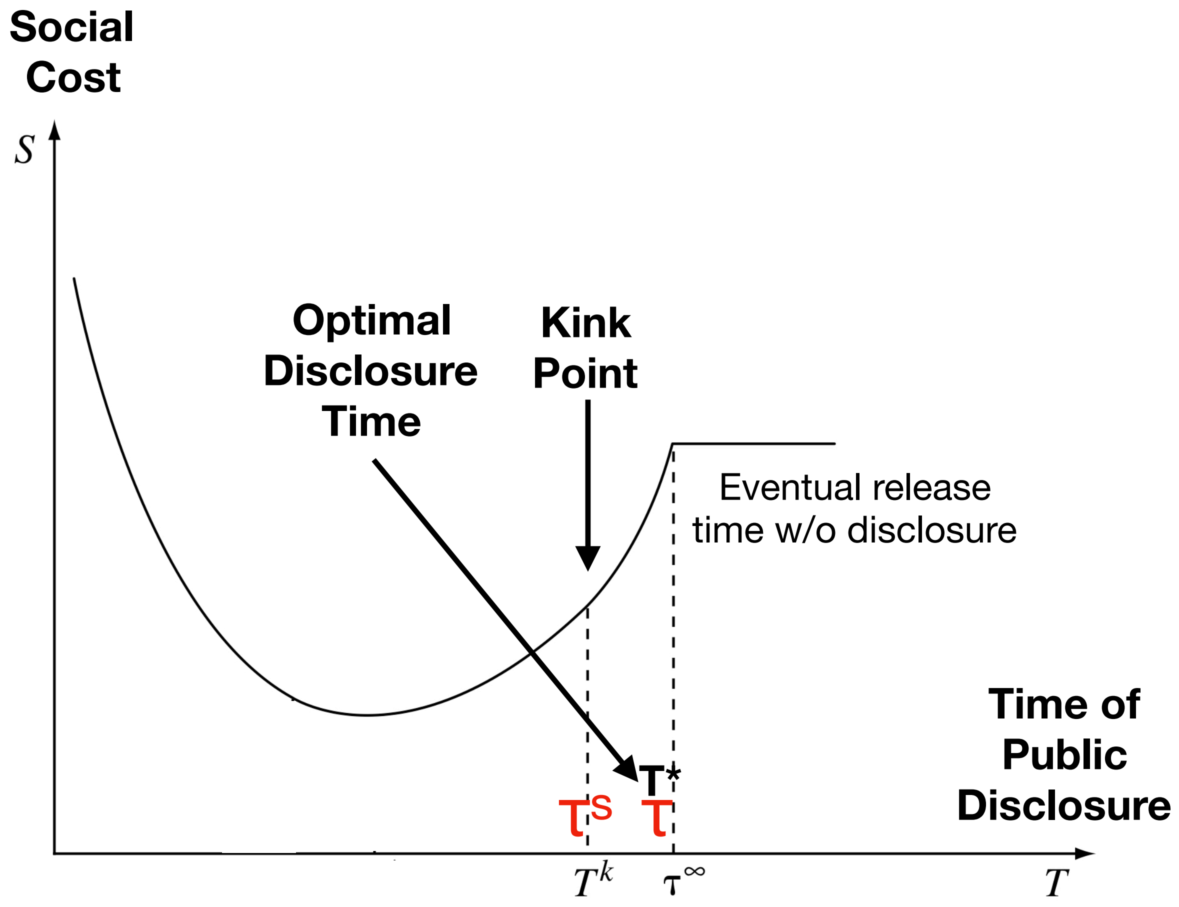
THEOREM 1. *For $T \in [0, T^k)$, the vendor patches after disclosure—i.e., $T < \tau^* < T^k$—and the slope of $\tau^*(T)$ is strictly less than one. For $T \in [T^k, \tau^\infty]$, the vendor patches at T—i.e., $\tau^* = T$—and hence slope of $\tau^*(T)$ is equal to one. For $T \in [\tau^\infty, t^{\text{end}}]$, the vendor patches at τ^∞, and hence the slope of $\tau^*(T)$ is equal to zero.*

Optimal Decision: Social Planner

- The social planner wants to minimise both the vendor's cost and the full user loss.
- If the vendor fully internalises user loss, it will release at τ^s, which may be before or after the kink point.
- If τ^s lies **after** the kink point (i.e. low cost or low vendor internalisation), social and vendor's optimal release times coincide.

The social planner's total social cost

$$S(T) = C(\tau(T)) + L(\tau(T), T). \tag{5}$$



THEOREM 3. *There exists a $\lambda^0 \in (0, 1)$ such that for $\lambda \geq \lambda^0$, $\tau^s \geq T^k$, and $\tau^* = T^* = \tau^s$ and T^* is independent of λ. For $\lambda < \lambda^0$, $\tau^s < T^k$ and the socially optimal protected period, T^*, is decreasing in λ.*

Optimal Decision: Social Planner

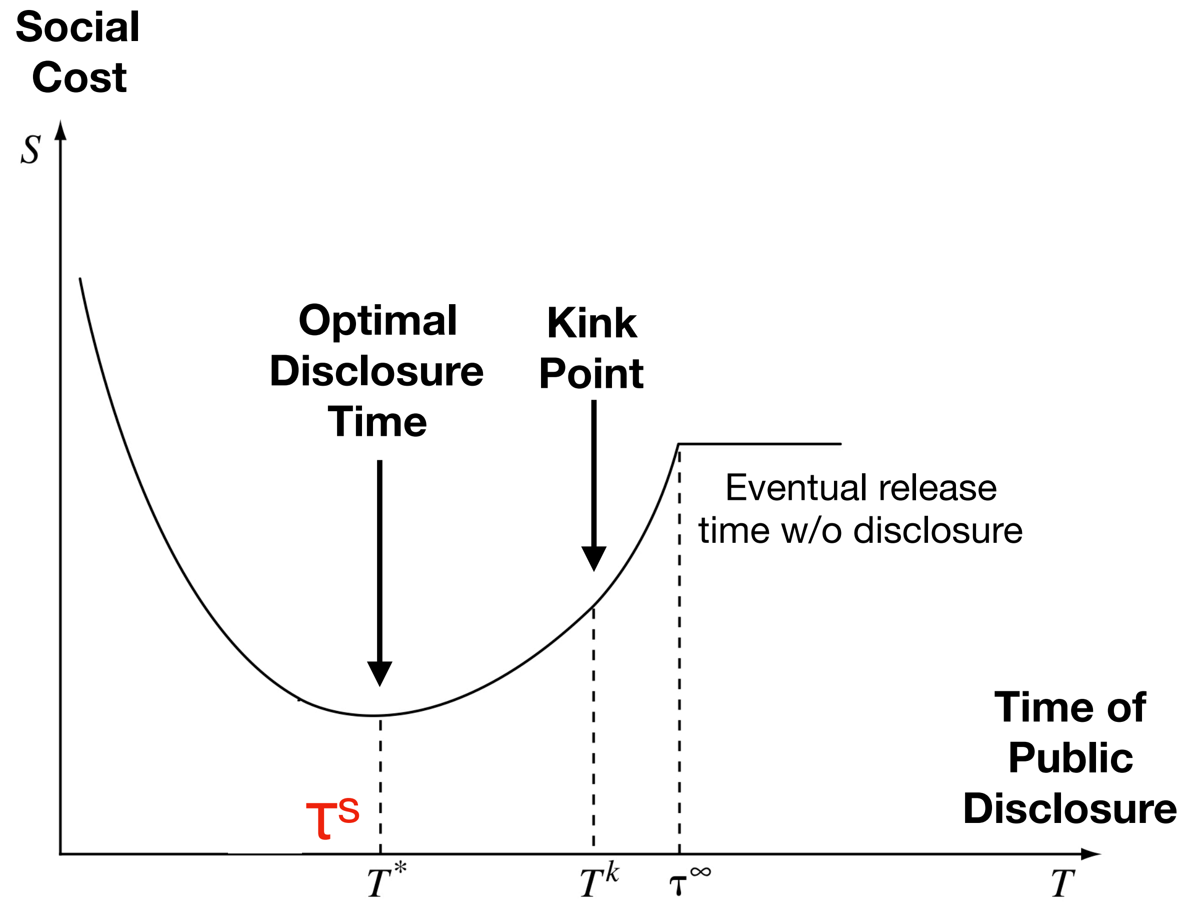
- If the vendor significantly internalises user loss (e.g. high risk vulnerability with potential damage to brand), $\boldsymbol{\tau^s}$ may lie **before** the kink point.

- In this case, the socially optimal release time can be chosen between $\boldsymbol{\tau^s}$ and the kink point.

- The social planner also has greater leverage over the vendor in this case.

The social planner's total social cost

$$S(T) = C(\tau(T)) + L(\tau(T), T). \qquad (5)$$



THEOREM 2. *When $\tau^s < T^k$, the socially optimal protected period T^* is bounded within (τ^s, T^k), i.e., $\tau^s < T^* \leq \tau(T^*) \leq T^k$. When $\tau^s \geq T^k$, $T^* = \tau(T^*) = \tau^s$.*

Optimal Decision: Variations

- The social planner issuing security warnings without disclosing the vulnerability can still cause vendor losses, and pushing the vendor to patch sooner.

- Post-disclosure loss can be higher than expected.

- The social planner may need to consider the vendor's practical needs in patch development when setting a disclosure date.

Modified vendor loss

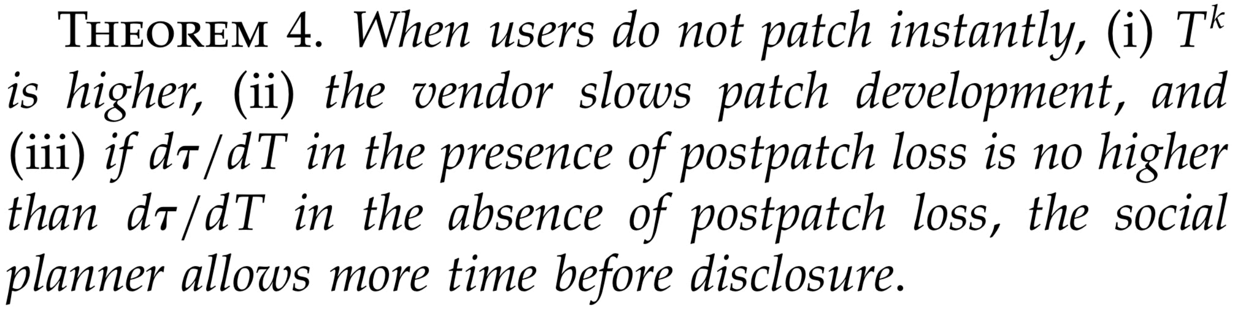
$$V(\cdot) = C(\tau) + \lambda L(\tau,\ T) + \psi(\tau,\ T)$$

Patch Quality and Postpatch Loss

- Customers may not apply a patch immediately, due to the patch's quality or mode of distribution.

- If the vendor always produces the same quality of patch, then both the social planner and the vendor pursue a less aggressive timeline.

$$
\begin{aligned}
V(\tau, T) &= C(\tau) + \lambda \mathscr{L}(\tau, T; q) \\
&= \underbrace{C(\tau)}_{\substack{\text{Vendor} \\ \text{Cost}}} + \underbrace{\lambda L(\tau, T)}_{\substack{\text{Pre-patch} \\ \text{Loss}}} + \underbrace{\lambda \tilde{L}(\tau; q)}_{\substack{\text{Postpatch} \\ \text{Loss}}}.
\end{aligned}
$$



Theorem 4. *When users do not patch instantly,* (i) T^k *is higher,* (ii) *the vendor slows patch development, and* (iii) *if $d\tau/dT$ in the presence of postpatch loss is no higher than $d\tau/dT$ in the absence of postpatch loss, the social planner allows more time before disclosure.*

Patch Quality and Postpatch Loss

- If given more time, the vendor can produce a higher quality patch, then a suitable amount of extension in disclosure time may be socially beneficial to a limited level.
- The beneficial effect diminishes over extended protection time.
- This depends on the levels of growth in development cost and postpatch loss.

$$\begin{aligned} V(\tau, q) &= C(\tau, q) + \lambda \mathscr{L}(\tau, T, q) \\ &= \underset{\underset{\text{Vendor Cost}}{\uparrow}}{C(\tau, q)} + \underset{\underset{\text{Pre-patch Loss}}{\uparrow}}{\lambda L(\tau, T)} + \underset{\underset{\text{Postpatch Loss}}{\uparrow}}{\lambda \tilde{L}(\tau, q)}. \end{aligned}$$

THEOREM 5. *When patch quality is endogenous, the optimal time for patch development* $\boldsymbol{\tau}^*$ *is increasing in disclosure time* T*, i.e.,* $d\boldsymbol{\tau}^*/dT > 0$*. If* $V_{\tau q}(\cdot) \leq 0$*, patch quality is increasing in T,* $(\partial q/\partial T \geq 0)$*; and if* $V_{\tau q}(\cdot) > 0$*, patch quality is decreasing in* T*,* $(\partial q/\partial T \leq 0)$.

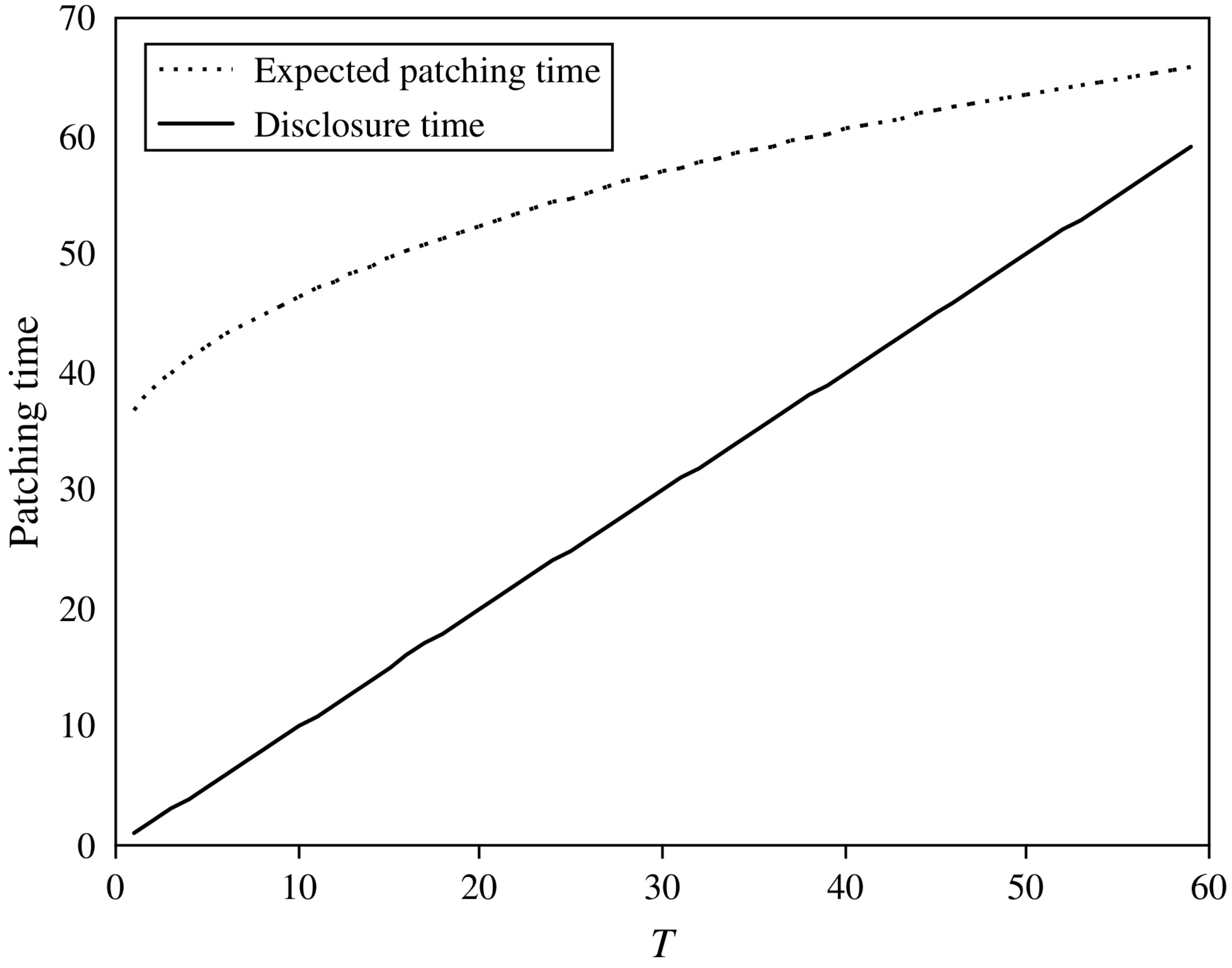
Numerical Simulation

- The numerical model tested through simulation on both low and high user losses. Data from empirical study were used in simulation.
- The optimal disclosure policy shown to be effective in reducing vendor's patch release time.
- Optimal disclosure policy is also more effective under higher user loss.
- Instant disclosure performed poorly in social cost as expected.

Empirical Data

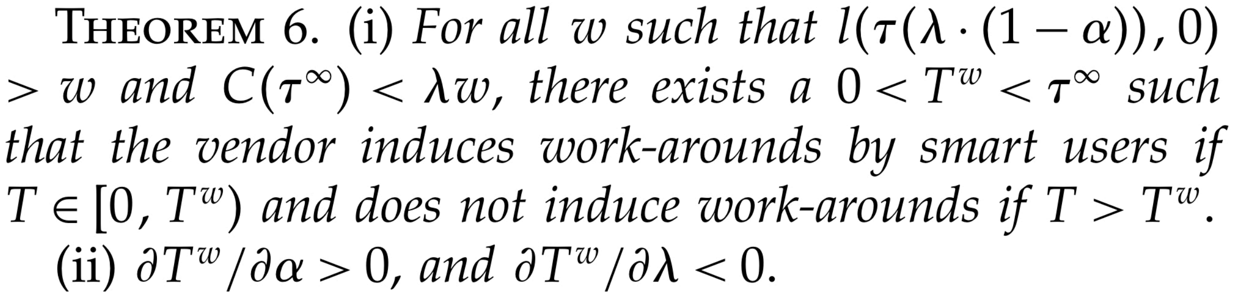
- The model was in general agreement with empirical data (Arora et al., 2005).

- Because in practice CERT may communicate with vendors directly, matching release and disclosure times were common.



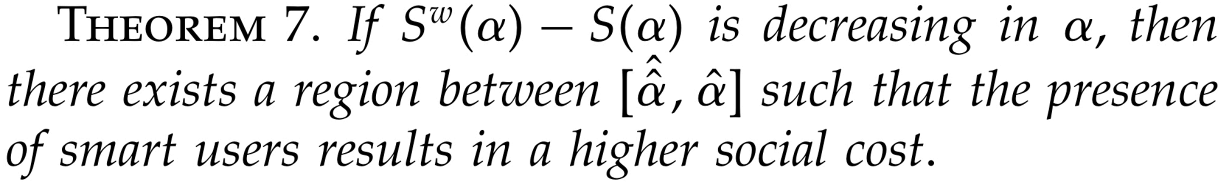
Customers Workarounds

- The social planner can theoretically inform technical customers of essential information to mitigate the vulnerability themselves.
- Technical customers receive a lower pre-patch loss, at the cost of creating an externality for customers relying on a patch.
- This also reduces vendor's incentive to release a patch quickly.



Theorem 6. (i) *For all w such that $l(\tau(\lambda \cdot (1-\alpha)), 0) > w$ and $C(\tau^{\infty}) < \lambda w$, there exists a $0 < T^{w} < \tau^{\infty}$ such that the vendor induces work-arounds by smart users if $T \in [0, T^{w})$ and does not induce work-arounds if $T > T^{w}$.*

(ii) *$\partial T^{w}/\partial \alpha > 0$, and $\partial T^{w}/\partial \lambda < 0$.*



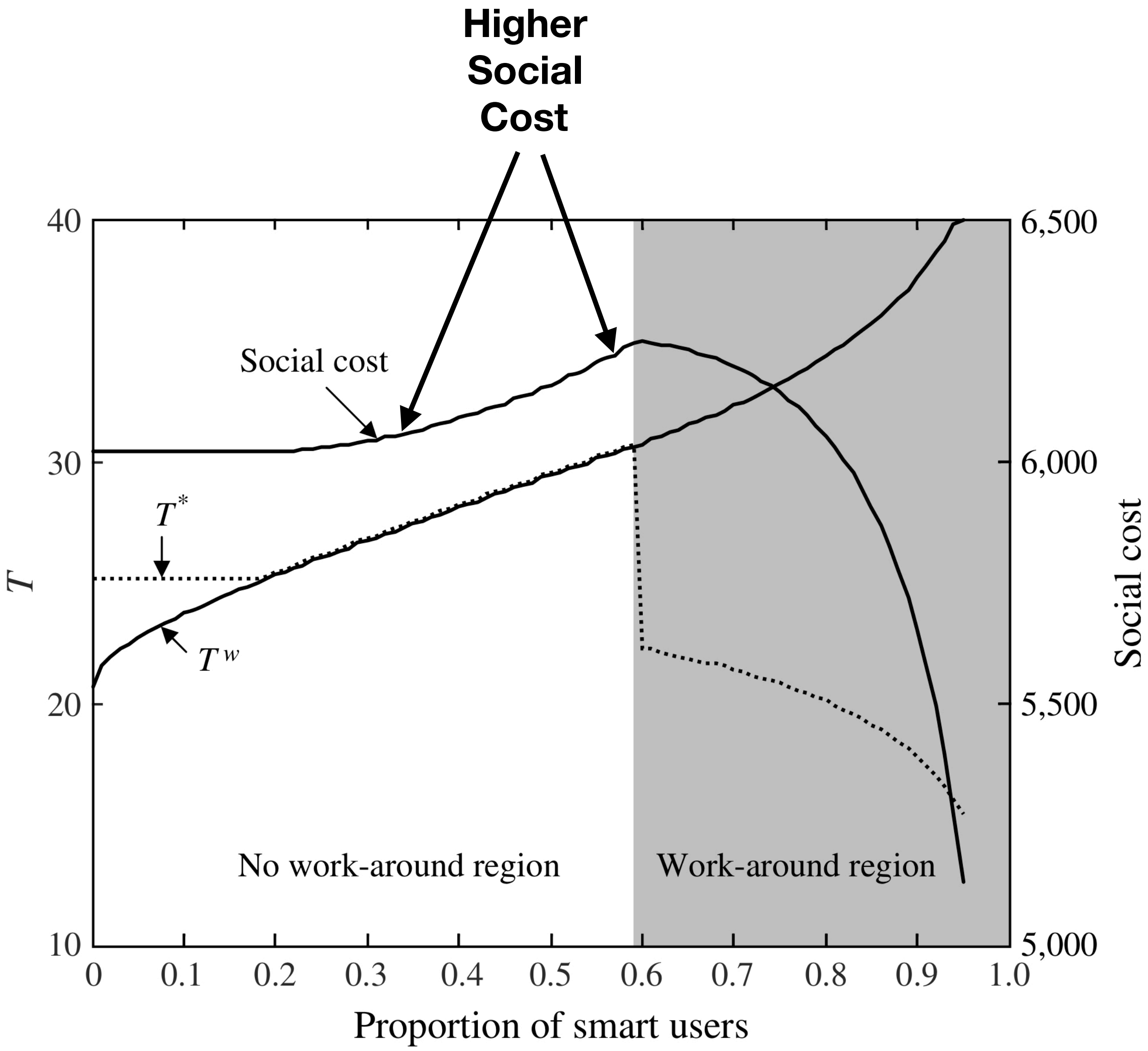
Theorem 7. *If $S^{w}(\alpha) - S(\alpha)$ is decreasing in α, then there exists a region between $[\hat{\hat{\alpha}}, \hat{\alpha}]$ such that the presence of smart users results in a higher social cost.*

Careful With Workarounds

- Under the model, a low internalised user loss and a higher proportion of technical users may convince the social planner to induce a workaround, as this minimises overall social cost if the vendor sets a long patch development time.

- This has the adverse effect of creating a negative externality for customers not being disclosed the workaround.

- Depends on the proportion of technical users, the workaround may result in a higher social cost.


Higher Social Cost

Current Context - EternalBlue

- Remote arbitrary code execution vulnerability in Windows SMB Server, used in secrecy by NSA for many years.

- Leaked in approximately August 2016, and disclosed to Microsoft.

- Time from vendor notification to public disclosure: >180 days, significantly longer than typical protection periods in the model.

- As Microsoft disclosed the vulnerability at patch time, $\tau^s = T^* = \tau^*$. No significant evidence of exploitation prior to disclosure despite the leak, low pre-patch loss.

- Reasonable quality patch distributed through Windows Update, but due to cooperative IT delays and such, massive ransomware outbreak happened 60 days later, signalling significant postpatch loss.

Critique

- The model performed a game theory style analysis of behaviours of the social planner and the vendor in vulnerability disclosure. The model roughly matched trends in the authors' own empirical data (Arora et al., 2005), which included fewer than 1000 disclosed vulnerabilities between 2000 and 2003. More up-to-date and broader data is required for further validation.

- Disclosing workarounds to technical users can provide wide ranging clues of attack vectors to would-be attackers, as there is no practical possibility of keeping that secret between a large proportion of technical users.

Suggested Discussion

- Serious vulnerabilities are often directly disclosed to the vendors, who then contact social planners such as CERT to allocate CVE numbers, as it has been the case for Meltdown and Spectre. What effects does this imbalance of information have on the assumptions and optimal policies of this model?

- Rather than disclosing workaround information to technical users, hardware vulnerabilities can be instead disclosed under embargo to operating system developers for them to develop patches in time for disclosure. This introduces multiple vendors into the model, which the paper suggested as future work. What practical additions to the study can better model multiple vendors?

References

Telang, Rahul, and Sunil Wattal. "An empirical analysis of the impact of software vulnerability announcements on firm stock price." IEEE Transactions on Software Engineering 33.8 (2007): 544-557.

Anderson, Ross, et al. "Measuring the cost of cybercrime." The economics of information security and privacy. Springer, Berlin, Heidelberg, 2013. 265-300.

Anderson, Ross, and Tyler Moore. "Information security: where computer science, economics and psychology meet." Philosophical Transactions of the Royal Society of London A: Mathematical, Physical and Engineering Sciences 367.1898 (2009): 2717-2727.

Arora, Ashish, et al. "Competitive and Strategic Effects in the Timing of Patch Release." WEIS. 2006.

Arora, Ashish, et al. "An Empirical Analysis of Vendor Response to Disclosure Policy." WEIS. 2005.