Incentive-Centered Design for Information Security

Rick Wash
School of Information
University of Michigan
rwash@umich.edu

Jeffrey K. MacKie-Mason
School of Information
University of Michigan
jmm@umich.edu

Abstract

Humans are "smart components" in a system, but cannot be directly programmed to perform; rather, their autonomy must be respected as a design constraint and incentives provided to induce desired behavior. Sometimes these incentives are properly aligned, and the humans don't represent a vulnerability. But often, a misalignment of incentives causes a weakness in the system that can be exploited by clever attackers. Incentive-centered design tools help us understand these problems, and provide design principles to alleviate them. We describe incentive-centered design and some tools it provides. We provide a number of examples of security problems for which Incentive Centered Design might be helpful. We elaborate with a general screening model that offers strong design principles for a class of security problems.

1 Introduction

People are the weakest link in security [1]. People write their passwords on sticky notes on the screen. People don't patch their home systems and become botnet zombies. People choose whether to label a patch "critical" or just "recommended." These actions generally reflect *motivated behavior* in response to the configuration of incentives confronting individuals.[1]

Incentive centered design (ICD) is a research area with the aim of designing systems that respect motivated behaviors, by providing incentives to induce human choices that improve the effectiveness of the system. ICD proceeds from rigorous mathematical modeling of strategic interactions between people (and their systems), to practical principles for system and protocol design. ICD differs from other economics of security work in its focus on providing concrete design principles.

The design of technology systems can have a great influence over the incentives that people have to use those systems. For example, in the early days of the commercial Internet there was a debate over whether the Internet should be application-blind, allowing anyone to put anything on it, or application-aware like cable TV and online services like AOL, with central authorities filtering content [8]. ICD modeling explained why the technical architecture of application-aware networks tends to limit the number of information goods offered, and biases selection toward mass market goods. This design issue is activated again in the current "net neutrality" policy debate.

When describing how a system works we include humans as smart, distributed and — crucially — autonomous components, with their own information sets and motivations. We draw primarily on microeconomics, game theory and cognitive psychology to model incentives, individual responses to them, and inter-individual strategic awareness and behavior. Because humans are non-programmable components, we often supplement mathematical and numerical model validation methods with human subject experiments.[2]

Much ICD research has focused on two problems pivotal to information security: *getting the good stuff in* and *keeping the bad stuff out*. We describe some examples in Section 2. For instance, individuals often are not directly compensated for the benefit their actions provide to others. Using a worm-throttling technology [14] on the border of my network benefits others, but I may have little incentive to install it because it only aids others after a worm has already penetrated my subnet. How can we design this technology so administrators are motivated to use it? This is the problem of getting the good stuff in.

Keeping the bad stuff out is similar to pollution. It arises when an individual does not bear the direct costs that her actions impose on others. When a spy places spyware on my machine, she uses CPU and bandwidth that degrade my use of the machine and she imposes other costs by appropriating my private information. The spy doesn't take these costs into account when choosing to distribute her wares. ICD focuses attention on behavioral incentives to discover who the polluters are (they

don't want to announce it!) and to discourage their polluting activities.

It is not possible to usefully summarize the full range of ICD models in a short paper. Instead, as in Section 3 we provide a more detailed discussion of a principal-agent model of screening, which can be applied to many hidden action and hidden information problems. The principal/designer sets terms and conditions for interaction with agents, who have their own objectives, and whose conduct may help or harm the principal. A successful screen provides incentives for agents to reveal their differences, so the principle can keep the bad stuff out.

2 Incentive Problems in Security

Incentives issues are implicated in many information security problems. We describe a few illustrative problems drawn from two interesting, but not exhaustive categories.

2.1 Getting the Good Stuff In

Labeling Vulnerabilities When announcing new vulnerabilities, vendors and security providers sometimes label them (e.g., "critical") and suggest that users and system administrators use the label to determine urgency and effort. This is a clear case of an ICD problem known as *hidden information*. The vendor *knows* more about these vulnerabilities than the users do, but in ICD we inquire into the *credibility* of the announced labels. Reporting a critical vulnerability in software makes the software look bad; reporting many critical vulnerabilities is even worse. This bad PR is a cost borne by the vendor but not the end users. There may be offsetting benefits from honesty, but the trade-off implies that vendors sometimes will underreport vulnerability severity.

Microsoft recently included the patch for an old vulnerability in a security update without disclosing this fact [12]. Two weeks later Microsoft released an emergency alert to install the patch because a dangerous worm exploit of this bug was published, and many systems had not installed the earlier under-labeled patch [11]. An ICD perspective could perhaps design a reputation service that provides vendors with sufficient incentives to provide more informative vulnerability labels.

Knowledge Workers In any sizable organization, knowledge workers make daily decisions that affect the security of the organization's information infrastructure. Does Bob leave his password on a sticky note under the keyboard, or memorize it? Will Ted just email the document instead of using the access-controlled storage system?

Careful attention must be paid to incentives for knowledge workers. Conflicts of interest between owners and employees are known in the ICD literature as *principal-agent* problems [9]. In this example there is a problem of *hidden action*: it is costly or impossible for the organization to perfectly monitor security compliance by employees. Much is known about design for such problems. For example, since appropriate security behavior is difficult to verify, incentives should be applied to other observable actions (proxies) that are closely linked with appropriate security behavior, with intensity of incentive positively correlated with the informativeness of the proxies. However, when there are multiple types of hidden action in which the organization has an interest (e.g., security compliance, mental effort, attendance to work rather than personal communications, diligence, etc.) incentives must be delicately balanced or the employee will favor some activities over others. For example, if bonuses are more dependent on timely project completion than on discovered security failures, the employee may overinvest in expedience at the expense of good security hygiene.[3]

Botnets Botnets, or networks of hacked machines under the control of a single attacker, have been used for a number of malicious purposes [13], including distributed denial-of-service attacks, and for sending spam and phishing emails. Botnets are possible because a large portion of computer users do not provide adequate security for their machines, either because they haven't been sufficiently motivated to learn how, or to care enough.

When a typical home user's computer is compromised for use in a botnet, he actually suffers very little; he possibly notices some system instability and network unreliability, but can easily attribute it to other external causes, such as normal variations in quality of service. He has little incentive to prevent such compromises, or to fix them once discovered, particularly since fixing the problem frequently involves reinstalling the operating system at great inconvenience. These users do not directly bear the costs of the victims of these botnet attacks, such as the downtime and lost sales for eBay. This has characteristics of an ICD problem known as the *private provision of public goods*, and earlier illustrated by the administrators who aren't motivated to install worm throttles for the benefit of external networks. What can be done to motivate individuals to contribute more effort and resources to the public good?

Privacy-enhancing technologies Numerous technologies "enhance privacy" by providing some level of anonymity. In general, these technologies work by making it very difficult to tell which node of a large set of nodes originated a communications. To work effectively

these systems require many traffic-relaying participants [5]. However, they suffer from a common incentive problem: they rely on nodes being willing to forward anonymous messages on behalf of others. Forwarding costs bandwidth and incurs risks, since anonymous communications are often anonymous for a reason. The rational choice may be to free-ride, using the system to send messages without contributing to it by forwarding others' messages, resulting in underprovision of forwarding nodes.

2.2 Keeping the Bad Stuff Out

Spam Spam (and its siblings spim, splog, spit, etc.) exhibits a classic hidden information problem. Before a message is read, the sender knows much more about its likely value to the recipient than does the recipient herself. The incentives of spammers encourage them to hide relevant information from the recipient whether or not the email is spam to try to get through the technological and human filters.

While commercial spam is not a traditional security problem, it is closely related due to the adversarial relationship between spammers and email users. Further, much spam carries security-threatening payloads: phishing and viruses are two examples. In the latter case, the email channel is just one more back door access to system resources, so spam can have more than a passing resemblance to hacking problems.

Spyware An installer program acts on behalf of the computer owner to install desired software. However, the installer program is also acting on behalf of its author, who may have different incentives than the computer owner. The author may surreptitiously include installation of undesired software such as spyware, zombies, or keystroke loggers. Rogue installation is a hidden action problem: the actions of one party (the installer) are not easy to observe. One typical design response is to require a bond that can be seized if unwanted behavior is discovered (an escrowed warranty, in essence), or a mechanism that *screens* unwanted behavior by providing incentives that induce legitimate to take actions distinguishable from illegitimate installers.

3 Screening

One problem of keeping bad stuff out arises when someone has a resource and wants users to have access to that resource, but has difficulty discerning good from bad users. The users know if they are "good" or "bad" (their type), so this is a problem of hidden information. This scenario characterizes phishing and spam, with good users being normal email senders and bad users as spammers or phishers. But it also covers less obvious situations. Are all automatic software installs and updates beneficial, or do some contain spyware? Or, are users attempting to login authorized (good) or unwanted hackers (bad)?

We illustrate with the last example, a well-known problem with a well-known solution. We do not here propose a new solution, but show how the known solution (passwords) can be understood as a screening mechanism. We offer this example to motivate the use of ICD theory, which provides design principles to generate screening mechanisms, to generate new or improved solutions for old and new problems.

Suppose a person, Principal, has a networked computer or other similar resource. Two users who we call Good and Bad have requested access to this resource. Principal knows that one of these users is a hacker, but cannot tell which one. He wants to provide access only to Good. To screen (differentiate) between them, he asks both to perform a task such as providing the correct password. This task is an effective *screen* if it induces Bad to act differently than Good. Once Principal can tell the difference, he can refuse restrict access by Bad. We now characterize the incentive properties that such a task must have to be effective.

More formally, let there be two user types, Good and Bad, indexed by θ^G and θ^B, with $\theta^G > \theta^B$. To access, User θ must perform a task with intensity $t \in [0, \infty]$. The task could, for example, be the provision of a password, a cash payment, or some work (e.g., requiring CPU cycles). The intensity t could represent the amount of payment or work required; e.g., in the case of passwords, the number of correct bits required. If User accesses the resource, Principal receives benefit $r(\theta^i, t)$ with $r_\theta > 0, r_t \geq 0$ (subscripts denote partial derivatives). This benefit r represents an aggregate of all the value the Principal gains from access by User, not just money. Principal's benefit is increasing in the quality of the user (θ) and the screening task performed by the user may be productive work ($r_t > 0$) or dissipative ($r_t = 0$, "make work"). One way to accommodate malicious users would be to specify that Bad who perform no task provide Principal with benefit $r(\theta^B, 0) < 0$. To attempt access costs a user $\alpha(t, \theta)$ (characterized below); when access occurs, a user receives a benefit of s provided by Principal.[4]

Suppose Principal knew in advance each user's type. If so, then Principal could offer one of two optimal "contracts", $\{(s^B, t^B), (s^G, t^G)\}$, where contract (s, t) states that if a user performs task t, Principal will provide sufficient access for benefit s. For example, Bad might be granted fewer resources (low s^B) or be required to provide substantial work in return (high t^B). A truly un-

desirable user (e.g., a malicious hacker) could be denied access (benefit $s = 0$).

In practice the user type is not known in advance by Principal. By offering an appropriate *choice* of contracts $\{(s^i, t^i)\}$ (determined below) Principal may be able to screen users so they, acting in self-interest, self-select into different contracts and thus reveal their type.[5] Users self-select by choosing at what level t to perform the task, receiving benefit s^i when they successfully perform task t^i. Determining these contracts is our design goal. By the Revelation Principle [10], we can, without loss of generality, restrict the set of contracts we consider to those in which the user finds it in her interest to truthfully reveal his type.[6]

Principal chooses the menu of contracts to maximize his total benefit subject to constraints, such as a budget constraint (possibly zero) on the screening cost. If λ is the fraction of users expected to be Good, Principal designs contracts to maximize his benefit by solving

$$\max_{\{(s^B,t^B),(s^G,t^G)\}} \lambda[r(\theta^G, t^G) - s^G] + (1-\lambda)[r(\theta^B, t^B) - s^B] \qquad (SCREEN)$$

$$\begin{aligned}
\text{s.t. } s^B - \alpha(t^B, \theta^B) &\geq u^0 && (1)\\
s^G - \alpha(t^G, \theta^G) &\geq u^0 && (2)\\
s^G - \alpha(t^G, \theta^G) &\geq s^B - \alpha(t^B, \theta^G) && (3)\\
s^B - \alpha(t^B, \theta^B) &\geq s^G - \alpha(t^G, \theta^B). && (4)
\end{aligned}$$

(1) and (2) are *Participation Constraints* (PC): a user must receive at least u^0 from making access to Principal's resource, or will choose not to participate (attempt access), for some u^0 determined by the user's other opportunities. (3) and (4) are *Incentive Compatibility* (IC) constraints: The payoff from truthfully revealing type must be greater than the payoff from dissembling to obtain the other type's treatment. The Revelation Principle allows us to impose the IC constraints to reduce the search space for an optimal solution.

We have not specified yet $\alpha(t, \theta)$, the cost of performing a task with intensity t for user type θ. Suppose the α can be constructed to satisfy the following conditions:

$$\begin{aligned}
\alpha_t(t, \theta) &> 0, \quad \text{for } t > 0 && (5)\\
\alpha_{tt}(t, \theta) &> 0 && (6)\\
\alpha_\theta(t, \theta) &< 0 && (7)\\
\alpha_{t\theta}(t, \theta) &< 0, \quad \text{for } t > 0. && (8)
\end{aligned}$$

We show below that these constraints produce an effective screen that will cause users to truthfully reveal their types. The first two conditions ensure the cost of performing the task is convex. Convexity means that task difficulty (cost) is increasing in t and at an increasing rate; this condition is sufficient though not necessary for the main results below. The third condition is that for any given level of screen t, the screen costs more for the Bad than Good users. The final condition is a single-crossing property we use below.

To illustrate, these properties are satisfied by standard password systems. Multi-bit passwords satisfy the convexity requirement (5), (6): the number of possible passwords is exponential in the number of bits. Providing a valid multi-bit password costs much less for Good (condition (7) because Good created or was told the correct password in advance and must merely retrieve it from storage, whereas Bad must use costly resources to guess it, including perhaps nontrivial timeout waits after making multiple incorrect guesses. The single-crossing property (8) requires here that the incremental cost of harder tasks is higher for Bad than for Good: the password storage and retrieval cost for Good is approximately linear, but the guessing cost for Bad is approximately exponential.

Define u^i to be the total net benefit (*utility*) to a user of type i; mathematically $u^i = s - \alpha(t, \theta^i)$. The solution of Principal's problem (SCREEN) yields several illuminating results:

Result 1 *Utility for Bad is minimal:* $u^B = u^0$.

Result 2 *Good gets a net gain:* $u^G > u^0$.

Result 3 *Good users receive more value from access* ($s^G > s^B$) *but perform a harder task* ($t^G > t^B$).

Results 1 and 2 are straightforward to prove; see, e.g., [6]. Result 2 has the following interpretation: Good owns valuable property — his knowledge that he is a Good type — and must be paid an *information rent* by Principal for the use of this property; the rent is $u^G - u^0 > 0$. Result 1 is a corollary: since everyone who doesn't prove they are Good is Bad, there is no reason to provide Bad with extra surplus to reveal her type.[7]

We demonstrate Result 3 with Figure 1. The curves $s - \alpha(t, \theta) = u$ are indifference loci (user utility is a constant for all (s, t) combinations on the locus), with utility increasing to the northwest (more s, less t). First consider the curve $s - \alpha(t, \theta^B) = u^0$ (the indifference locus for Bad); we know from Result 1 that (s^B, t^B) lies somewhere on this locus. Now construct Good's locus through (s^B, t^B); by (8) there is a single crossing and the slope is less. Since $\alpha_\theta < 0$, by (7), $s^B - \alpha(t^B, \theta^G) = u^1 > u^0$ (Good receives some surplus utility from choosing contract (s^B, t^B).) By (4), (s^G, t^G) lies to the southeast of the B locus $s - \alpha(t, \theta^B) = u^0$. By (3), (s^G, t^G) lies to the northwest of $s - \alpha(t, \theta^G) = u^1$. Thus, (s^G, t^G) must lie in the shaded area, and Result 3 obtains.

For the password example, the task t is to provide correct bits; when a password system works, Good provides more bits than does Bad, consistent with Result

Assumptions (5)–(8) represent properties of the task that ensure that screening will work, and hence are principles for the design of such tasks. One of the most important design principles is not a result, but assumption (8): the incremental cost of performing the screening task must be lower for Good than Bad types (the analogous principle holds for a continuum of types). This ensures that for a given payoff, a Good type will reveal himself by a greater willingness to perform the task. If the task is equally difficult for both Good and Bad types (not satisfying (8)), then it will not differentiate users. This design principle is only sufficient to ensure such contracts exist; they specifics for a given screen still must be found.

We illustrate the usefulness of Results 2-3 and assumption (8) by sketching their application to other "keep the bad stuff out" problems. Challenge-response systems such as CAPTCHAs are intended to prevent automated agents from hijacking various online resources. To get agents to reveal their type as human or bot, the task solving cost must be higher for the bot (presumably in CPU or programming time). Once revealed, the bot is denied access ($s^B = 0$), and its owner gets only the value from its next best alternative activity, u^0. Thus bots usually don't attempt to crack CAPTCHAs, satisfying Result 3 that the Good types exert more effort on the screening task (or, as with passwords, we could interpret t as the number of correct bits provided, which will be lower for bots if faced with an effective CAPTCHA screen). Of course, if Bad types find ways to make the incremental solving cost similar to Good's cost (violating (8), say through cheap automated pattern-matching, or cheap porn-bribes to human puzzle solvers, the screen will be ineffective.

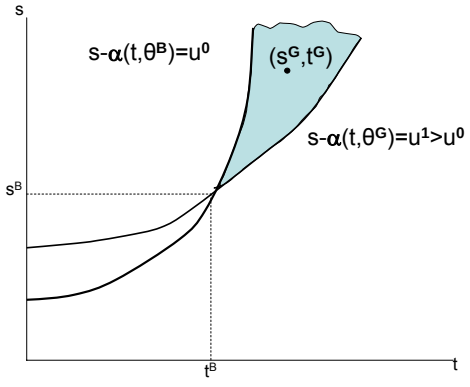

Figure 1: Screening good from bad

[4] proposed a challenge-response approach to spam reduction; [2] describe a method that illustrates screening theory. Senders must perform CPU-cycle-burning tasks to obtain a valid (personal) address for a recipient. The screening task may be more costly for spammers because competition requires them to run servers at full capacity, so a CPU task for every valid email address becomes prohibitive. Good agents are presumed to have machines sufficient idle cycles. The authors of [2] magnify the cost differential by allowing recipients to repudiate valid incoming email addresses if a single spam enters that channel, so spams incur the CPU cost to obtain a new address for nearly every message sent, while good senders only need to pay for a good address one time.

One of us proposed a related mechanism to fight unsolicited communication using repudiable cash bonds as the screen [7]. Senders put t in escrow; recipients can claim the bond or let it revert to sender. The bond cost is higher to bad types if they face a higher probability that the recipient will claim the bond; presumably recipients will not always claim bonds from good types because they want future communications from the good types.

4 Discussion

Many problems in information security exist at least partially because the people involved are not properly motivated to solve them. Incentive-centered design provides tools and principles to guide technology development for security systems. As an example, we developed a screening model and showed how the design principles it provides have been used in existing security technologies. The key insight is that human behavior — whether cooperative, indifferent or malicious — is not a fixed constraint. Rather, humans have goals, and choose their behavior to advance their goals. Design with this in mind can produce systems that change incentives, and thus harness behavior to advance the designer's goals. The incentive-centered design literature provides rigorous theories and methods for implementing this general approach.

Traditional information security generally deals with keeping the bad stuff out. It has been fairly successful at keeping hackers, viruses, and worms at bay. We described some more recent security problems in this category. The screening model above prescribes design features that induce people to self-identify, so resource managers can keep the bad stuff out. Many of the design principles from the screening model are intuitively understood by the people who develop security technologies. Password systems are an example of technology that got this right, and as a consequence, are — if correctly used

— effective at separating legitimate users from attackers. The same principles — or others from ICD, as screening is but one model from this field — can be applied to develop new or improved solutions for both new and old problems.

Even when technical security systems are effective at keeping the bad stuff out, they may fail to get the good stuff in. Passwords again are a good example: they often fail because users are insufficiently motivated to use strong passwords that prevent password guessing attacks. Incentive-centered design, can create systems that motivate users to provide desired security effort. Indeed, incentive-centered design, as an alternative to technological "hardening", may be especially effective in those applications involving agents who are not malicious, but merely undermotivated, since it is not their objective to thwart the system. Non-malicious agents may be more robustly responsive to incentives.

In future research we will model "getting the good stuff in" problems to provide design principles (and implemented designs) for this class of problems.

Acknowledgments

Our thinking was improved by conversations with Paul Resnick, Yan Chen, Rahul Sami, Brian Noble, Yoshi Kohno, Mark Ackerman, Marshall van Alstyne, and the rest of the ICD lab group at Michigan (Lian Jian, John Lin, Anna Osepayshvili, Kil-sang Kim, Nese Nasif, Greg Gamette, and Ben Chiao). Emilee Rader and Chris Connelly contributed greatly to readability.

References

[1] ANDERSON, R. Why cryptosystems fail. In *ACM First conference on Computers and Communications Security* (1993).

[2] BLEICHENBACHER, D., GABBER, E., JAKOBSSON, M., MATIAS, Y., AND MAYER, A. Curbing junk e-mail via secure classification. In *Proc. of the 2nd International Conference on Financial Cryptography* (1998), pp. 198–213. Available from `citeseer.ist.psu.edu/86989.html`.

[3] CHEN, Y., LI, X., AND MACKIE-MASON, J. K. Online fund-raising mechanisms: A field experiment. *Contributions to Economic Analysis & Policy 5*, 2 (2006). `http://www.bepress.com/bejeap/contributions/vol5/iss2/art4`.

[4] DWORK, C., AND NAOR, M. Pricing via processing or combatting junk mail. In *Crypto '92* (1992), pp. 139–147.

[5] GOLDBERG, I. *A Pseudonymous Communications Infrastructure for the Internet*. PhD thesis, University of California, Berkeley, 2000.

[6] LAFFONT, J.-J., AND MARTIMORT, D. *The Theory of Incentives*. Princeton University Press, Princeton, 2001.

[7] LODER, T., VAN ALSTYNE, M., AND WASH, R. An economic solution to unsolicited communications. *Advances in Economic Analysis and Policy 6*, 1 (2006).

[8] MACKIE-MASON, J. K., SHENKER, S. J., AND VARIAN, H. R. Service architecture and content provision: The network provider as editor. *Telecommunications Policy 20*, 3 (Apr. 1996).

[9] MAS-COLELL, A., WHINSTON, M. D., AND GREEN, J. R. *Microeconomic Theory*. Oxford University Press, New York, 1995.

[10] MYERSON, R. B. Incentive compatibility and the bargaining problem. *Econometrica 47* (1979), 61–74.

[11] NARAINE, R. 'detailed exploit' published for critical windows flaw. *eWeek.com* (June 26 2006). `http://www.thechannelinsider.com/article/Detailed+Exploit+Published+for+Critical+Windows+Flaw/181984_1.aspx`.

[12] NARAINE, R. Microsoft's security disclosures come under fire. *eWeek.com* (April 13 2006). `http://www.eweek.com/article2/0,1895,1949279,00.asp`.

[13] THE HONEYNET PROJECT AND RESEARCH ALLIANCE. Know your enemy: Tracking botnets. `http://www.honeynet.org/papers/bots/`.

[14] TWYCROSS, J., AND WILLIAMSON, M. M. Implementing and testing a virus throttle. In *Proceedings of the 12th USENIX Security Symposium* (2003).

Notes

[1]The *motivated behavior* framing is more general than it might seem at first blush. For example, security failures due to underinformed users might be investigated as a problem in a failure to provide incentives to be better informed. Not every human action can be analyzed as a rational response to incentives, but a surprising number yield usefully to this framework.

[2]For example, one of the authors recently conducted an online experiment in "getting good stuff in" to test multiple theories of motivating public contributions to an online library[3].

[3]The theory does not stop at these qualitative characterizations, but guides measurement and quantitative design.

[4]For simplicity we assume that a transfer worth s units of value to Principal is also worth s units to a user, such as would be true for a money payment. But the value transfer typically won't be a cash payment, but resource usage such as CPU cycles. It is straightforward to modify the analysis so that a transfer that costs Principal s is worth some function $f(s)$ to User.

[5]Contracts may be implicit; it is not necessary that they be formal.

[6]The Revelation Principle says that for any set of contracts under which a rational agent is not truthful, there exists another set of contracts under which a rational agent wants to report honestly, and the payoffs in all states of the world are the same as in the first set of contracts. The intuition is straightforward: for whatever map agents are using to transform their true type into an announced type in the original mechanism, imagine a different mechanism in which agents announce their true type, and the rules of the mechanism then apply the same map as the agents were using before to transform their truthful inputs into their dishonest inputs, and then assigns the outcomes they would have gotten for those dishonest inputs in the original mechanism.

[7]We have formulated the problem as if Bad can provide some positively value to Principal, albeit less than Good. If Bad types provide strictly negative utility to Principal (they are always malicious) no matter the amount of task t they perform ($r(\theta^B, t) < 0 \; \forall \; t$), then we could easily reformulate the problem with a non-participation constraint to replace (1) and a change in the Principal's objective function to correctly account for non-participation by successfully screened B types. The results above would be the same, and thus for discussion we assume that B types who are indifferent about participating ($u^B = u^0$) merely go away without attempting entry if the screen would be effective.