2007-03

Planning Cost-Effective Deceptive Resource Denial in Defense to Cyber-Attacks

Rowe, Neil C.

Monterey, California. Naval Postgraduate School

Planning Cost-Effective Deceptive Resource Denial in Defense to Cyber-Attacks

Neil C. Rowe
Cebrowski Institute, U.S. Naval Postgraduate School
Code CS/Rp, 833 Dyer Road, NPGS, Monterey CA 93943 USA
ncrowe@nps.edu

Abstract: Cyber-attacks against computer systems that provide valuable services can often be effectively defended by tactics of deliberately deceptive resource denial. Delaying in response to suspicious requests is one example; it permits time to develop a good defense, facilitates analysis of the attacks and formulation of a response, and may little affect legitimate users. But delays can look suspicious; a better tactic can be for the operating system to falsely claim unavailability of some critical resources that the attacker needs (files, directories, access rights, network connections, or software). This can be more effective than using "security policy" as an excuse to deny those resources because it is unexpected and more flexible. We formulate a decision-theoretic approach to the problem of deciding when to deceive by resource denial in a sequence of interactions with a user of an operating system, and provide general formulae for decisions in planning deceptions. Our theory covers both reactive and proactive deception, and both single-session and multi-session attacks. We also provide additional criteria to ensure logically consistent tactics. We provide some evidence from a survey of users to support our modeling.

This paper appeared in the 2nd International Conference in I-Warfare and Security, Monterey CA, USA, March 2007.

Keywords: Deception, cyberspace, decision theory, resources, denial, lies

1. Introduction

Rapid cyber-attacks can often be best defended by delaying tactics. Delaying facilitates analysis of the attacks and formulation of a response, particularly for a new kind of attack. This is valuable for systems like those of the critical national infrastructure that must be kept running. At the same time, delays will rarely affect legitimate users if they are only applied after suspicious behavior is observed. Delays can be accomplished by waiting to respond to commands, as in "wrappers" around operating-system commands that delay when informed of suspicious behavior by an intrusion-detection system (Rowe, 2004). But delays by themselves look suspicious; a more sophisticated delay is wasting of a user's time. This can be done by an operating system falsely claiming unavailability of resources that the attacker wants (files, directories, access rights, network connections, and software) at some time into a session. This can be more effective than denying access rights based on security policy because it works against insiders with privileges as well as outsiders, discourages less a repeat of the same (foiled) attack, and encourages underestimation of defenses by the attacker. They may later find they have been deceived, but it will take them some time. By then the system administrators can be aware of the attack, and vulnerable software and ports can be shut down.

False resource denial is a form of deception. Deception is an important human strategy and tactic to accomplish goals involving manipulation of people (Ford, 1996). Deception, used sparingly, is often more cost-effective social manipulation than direct attempts to influence people, as is attested to by its many dramatic uses in military history. However, deception has a price: People do not like being deceived. If they discover deception, they may react strongly and negatively. Furthermore, legitimate users of a computer system who are deceived about resource availability may be unable to accomplish their goals, so deception may increase the cost of doing legitimate business. Thus we need a model to analyze the tradeoffs involved in deception. While there are ethical concerns about deception, most ethical theories permit deception to prevent a significantly greater harm, and destruction of the software of a computer system is a serious harm.

After surveying previous work, we build a decision-theoretic model of online deception in section 3. We show how to introduce probabilities and costs (section 3.1), and argue for statelessness of the analysis (section 3.2). We then evaluate anticipatory deception (section 3.3) and multi-session deception (section 3.4). We then point out issues in logical consistency of deceptions (section 4) as a way to make them more convincing. We conclude with some observations and directions for future work.

2. Previous work

Deception is a common social phenomenon. People use deception frequently for worthwhile ends (Nyberg, 1993), and it would be impossible for societies to function without it in many areas such as law, politics, business, entertainment, and psychology. Attackers of computer systems use deception themselves in their identities and software tools, so it would seem fair to use deception in defending our systems too. Deception has the advantage of being a generally unexpected defense for computers, and one that can be difficult for attackers to perceive.

Deception has been used effectively for “honeypots”, computer systems intended solely for collection of data about attacks (The Honeynet Project, 2004). For defenders to collect useful data with a honeypot, the attacker must think that the system is an ordinary one. To this end, it helpful to create fake files and data on the honeypot to suggest that real users have been using it in normal ways (Cohen & Koike, 2003; Gerwehr et al, 2000). Such fake data can include passwords to other machines, credit-card numbers, and other things for which we can confirm use.

A variety of models of deception have been constructed for psychology (Heuer, 1982) and counterintelligence (Whaley, 1982). Recent computational models include the descriptive models of (Cohen, 1999), the linguistic speech-act model of (DeRosis et al, 2003), agent-based deception-planning models (Christian & Young, 2004), and reputation-systems assessment of deceptiveness (Barber & Kim, 2001). Our previous work built probabilistic models of belief in “generic excuses” for why the attacker could not achieve their goals, plus the belief of the attacker in the hypothesis "I am being deceived" which we do not want them to have (Rowe, 2004); excuses are a versatile strategy for refusing to do things (Snyder, Higgins, & Stucky, 1980).

3. Choosing if and when to deceive

In what follows we assume a computer system is under attack by a relatively alert attacker who is trying to achieve particular goals (Chirillo, 2001). These are reasonable assumptions for insider attacks, state-sponsored or organization-sponsored outsider attacks, and sophisticated hackers. At the same time, our deceptive responses will be unexpected events for any attacker, and thus will easily stop most automated or scripted attacks.

We can initiate deceptions about resource availability at many possible times in a sequence of commands. However, some opportunities are better than others because (a) some deceptions are more convincing, and (b) some deceptions cause less harm to legitimate users who are accidentally suspicious. We will use a decision-theoretic model because this has been effective in understanding deception in sociobiology (Lachmann & Bergstrom, 2004).

A range of strategies for using deception to defend computer systems are possible (Rowe & Rothstein, 2004). One major strategy is to try to “scare away” the attacker by making the system seem dangerous to attack. This can be done, for instance, by making a system appear to be a honeypot when it actually is not (a “fake honeypot”), because most attackers know honeypots record their actions and are difficult to exploit (Rowe, Duong, & Custy, 2006). Another strategy is to encourage the attacker to leave by making it seem like the attack target is not exploitable, as when key features necessary for their attack are absent. This can be done, for instance, by telling the attacker that the network is down so that they think they cannot download their attack tools. Lies like this about system resources are generally simple and easy to accomplish, and hard for an attacker to disprove. We investigate this second strategy in this paper.

3.1 Incorporating probabilities and costs

For the simplest model (Figure 1), we assume that the user is malicious with probability p_m, a malicious user will continue attacking the system after the deception with probability p_{mr}, and a legitimate user will continue working after the deception with probability p_l. The first will be obtained from an intrusion-detection system (Monteiro, 2003) and will vary based on user behavior. The others will be personality- and task-dependent because some users are just more determined than others (Lydon & Zanna, 1990). Assume also the cost of allowing an attack to succeed is c_m, and the cost of a legitimate user not achieving their goals is c_l. Costs can be time, but could include subjective factors like aggravation too.

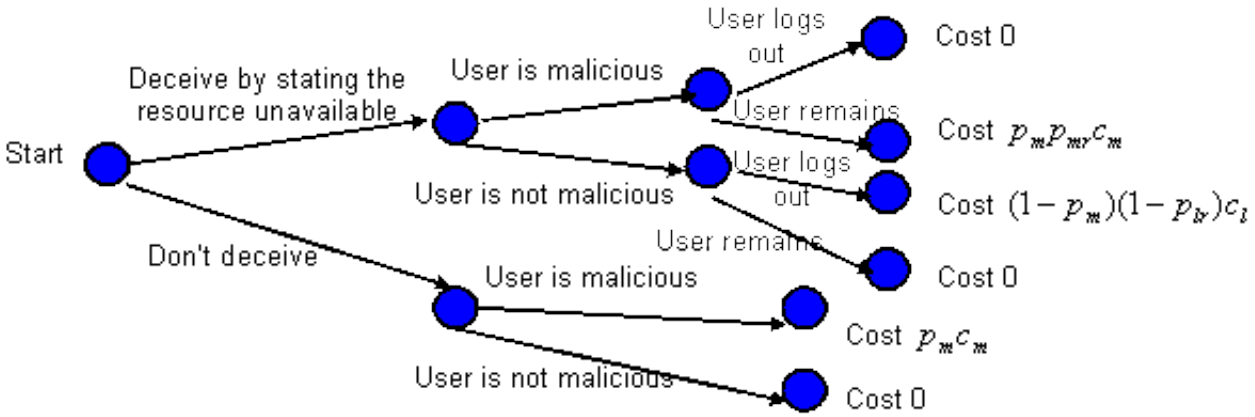

Figure 1: The simplest model of the decision to deceive at some point in a command sequence.

Then the expected cost of deceiving will be less than the cost of not deceiving if $p_m p_{mr} c_m + (1-p_m)(1-p_{lr})c_l < p_m c_m$.

If we define $g(x) = x/(1+x)$, this can be written as $p_m > g[c_l(1-p_{lr})/c_m(1-p_{mr})]$. This g is the function that converts odds to probabilities, so the expression is brackets is analogous to an odds measure. The inequality says that if a deception is completely ineffective in encouraging the malicious user to go away, it is never desirable; if it completely ineffective in encouraging the legitimate user to go away, it is always desirable; and if the deception no more affects the attacker's goals than the legitimate user's goals, the inequality reduces to $p_m > c_l/(c_l + c_m)$ so even then the deception may be desirable.

For example, suppose we lie that the network is down (nonfunctional) in response to an attack involving attempted rootkit installation. Assume the cost of a successful rootkit installation is 2 hours of administrator work to reinstall the operating system and fix bugs. Assume the cost to a legitimate user of the network being down is 5 minutes of work because they can copy to a portable storage medium to do local file transfers. Because network file transfer is essential to most rootkit installation methods, a malicious user will have a low probability of continuing their attack after this deception, say 0.1. On the other hand, legitimate users probably have other tasks not needing the network such as editing, so their probability of continuing is something like 0.5. Therefore $p_m > g[(5*0.5)/(120*0.9)] = 0.023$ and we should deceive if the user has at least 2.3% chance of being malicious.

We conducted a survey (see Appendix) to estimate some key probabilities for this analysis. The survey was administered to seven students and five faculty members at our school. Geometric means of the response values for each subject ranged from 0.187 to 0.000002, but *ratios* of responses between questions were much less variable, indicating that conditional probabilities were more consistent across subjects than a priori probabilities (which may reflect personality). The survey confirmed that deception was considered a possible explanation for rare events on a computer system, with conditional probabilities of 0.32, 0.54, and 0.09 for deception with the reported conditions "network down", "local network down", and "network messing up files". Thus subjects understood that while deception is rare a priori, it is not conditionally rare. Our subjects also were quite persistent in trying the same action again when it failed, so p_{mr} and p_{lr} should be close to 1; this suggests we can often use the approximate criterion $p_m > c_l/(c_l + c_m)$.

3.2 Refining the deception model

A useful tool is:

> *The Statelessness Assumption: The resources of a computer system remain constant in their availability status through a session of a user.*

This is often reasonable because most sessions are just minutes long. Then the above model implies we should wait as long as possible to deceive an attacker to be more sure that they are malicious. However, (Josang, 2001) suggests that

distrust grows more easily than trust; waiting gives the attacker time to become distrustful. (Rowe, 2004) proposes that the believability of a deception for an attacker is a product of the a priori likelihood of the deception event, the a priori likelihood that the system does not engage in deception, and the intrinsic suspiciousness of what the attacker has done previously. Thus attackers engaged in suspicious behavior are more suspicious of unusual events that thwart them on an important system. We will postulate that attackers recognizing deception will not log out because they figure they can find a way around it, whereas unsuspicious attackers will log out because deceptions like “network down” can be chosen to suggest impossibility of most attacker plans.

This gives us enough constraints to create a Bayesian model (Korb & Nicholson, 2004) of p_{mr}, the probability the malicious user remains logged in after a deception that would imply that they cannot achieve their goals. Using Bayes' Rule where d is the occurrence of a deception action and c is the condition that the attacker believes they are being deceived, $p(c \mid d) = p(d \mid c)p(c) / p_d$ and $p(\neg c \mid d) = p(d \mid \neg c)(1 - p(c)) / p_d$, so if $p_{mr} \approx p(c \mid d) / (p(c \mid d) + p(\neg c \mid d))$, $p_{mr} = g[p(c)p(d \mid c) / (1 - p(c))p(d \mid \neg c)]$. Thus for the deception "network down" and the probabilities $p(c) = 0.01, p(d \mid c) = 0.1$, and $p(d \mid \neg c) = 0.01$, we estimate $p_{mr} = 0.9$. On the other hand, we assume p_{lr} is independent of circumstances because many deceptions can be chosen to be in features inessential to a legitimate user.

The p_{mr} can also be calculated even when statelessness is not assumed. For example, the condition "network down" is around 0.02 on computer systems the author uses; but if the network was down one hour ago, it is likely to be down now. A down period for the author's network has an expected duration of one day. In general, we can use a Poisson model much like that for M/M/1 queuing models, where λ_D is the number of times that the condition D would legitimately occur sometime during a day (the arrival rate for a queue), and λ_U is the number of times that D would be remedied during a day (the service rate). Then if we reported to the user that D was true at some time, the probability that D is still true at a time t units later is $p_U = e^{-\lambda_U t}$; if we reported to the user that D was false at some time, the probability that D is still false at time t is $p_D = e^{-\lambda_D t}$.

Another reason to avoid waiting too long to deceive is that the number of available deceptions decreases during a session as resource availabilities are confirmed if we maintain statelessness. For instance, "network down" is less convincing when the user has encountered unrelated software problems already. This is similar to the factor of “mobility” in chess, where of two positions with equal piece count and danger, the one with more possible moves is better because it is less likely to lead to a forced bad move. We can incorporate a mobility factor in our decision model if we choose.

3.3 Anticipatory deception

We now analyze anticipatory deception, where we deceive when we are mildly suspicious to permit a consistent deception later if necessary. For instance, we can say the network is down in response to an attempt to transfer an innocuous file, in anticipation of a later attempt to transfer a rootkit. The first deception enables consistency of the "network down" hypothesis and reduces attacker suspicion. If we deceive more than once, we should use the same deception each time because deceptions will be rare and two unrelated deceptions will be particularly suspicious. Thus generic excuses like "network down" are useful as deceptions because they explain failures of many separate things.

We now refine our decision model (Figure 2). Here we distinguish p_{m1}, the probability the user is malicious at the first deception, from p_{m2}, the probability at the second deception. We similarly distinguish p_{mr1}, the probability a malicious user will remain after the first deception, from p_{mr2}, the probability after the second. We also distinguish p_{lr1} from p_{lr2}, and both from p_{mr2d} and p_{lr2d}, the probabilities of the malicious or legitimate user remaining when the first deception is

not done but the second deception is done. We can assume that $p_{br1} = p_{br2} = p_{br2d} = p_{br}$ because if deceptions are designed with attackers in mind, each is equally likely to make a legitimate user give up and go away. We also assume that $p_{mr2d} > p_{mr2}$ and $p_{mr2d} > p_{mr1}$, because the "mr2d" case most obviously suggests deception, and apparent deception should make a malicious user want to remain. We can then calculate expected costs for three strategies for a particular deception D in the course of a short session (else include the exponential decay):

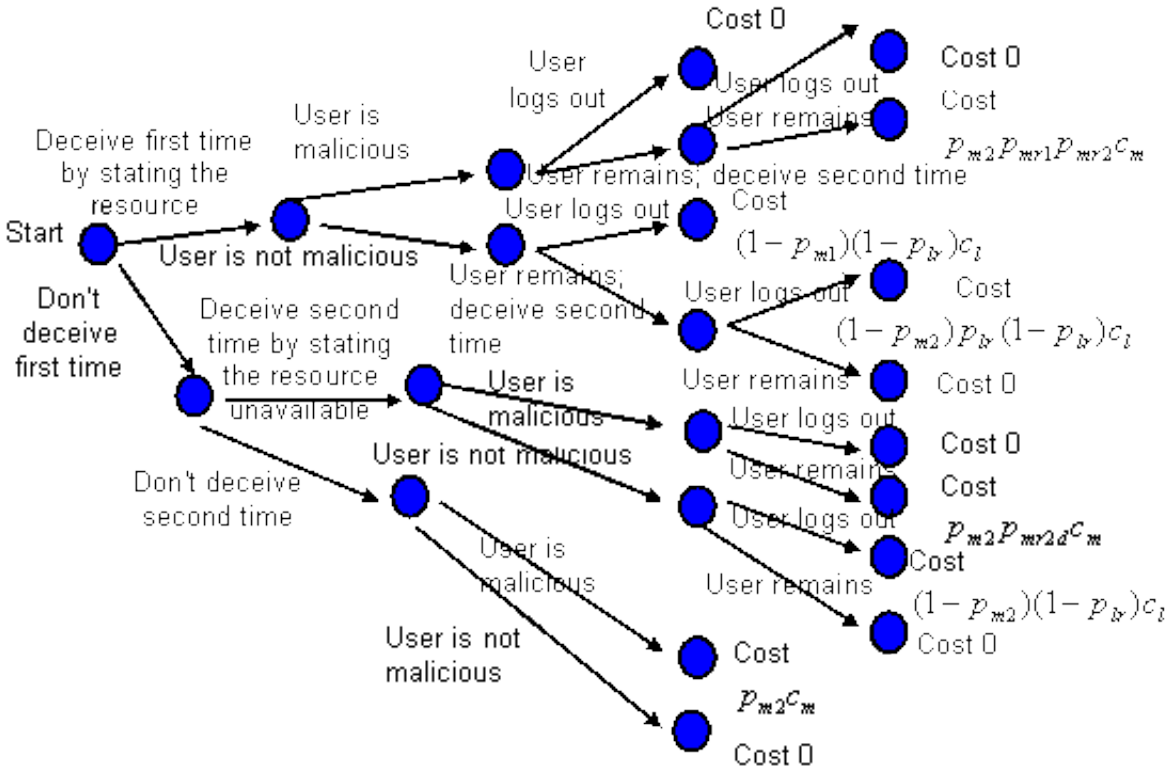

Figure 2: Analysis of anticipatory deception.

- S1: No deception: $p_{m2}c_m$
- S2: Deceive only at the second opportunity: $p_{m2}p_{mr2d}c_m + (1-p_{m2})(1-p_{br})c_l$
- S3: Deceive at both opportunities:

$$p_{m2}p_{mr1}p_{mr2}c_m + (1-p_{m1})(1-p_{br})c_l + (1-p_{m2})(1-p_{br})p_{br}c_l$$

Note it is useless to deceive only at the first opportunity, as this does not foil an action necessary to the attack, creates a suspicious inconsistency, and risks antagonizing legitimate users.

There are no simple formulas comparing the strategies, but there are special cases. As before, S2 and S3 will be suboptimal if the malicious user is never fooled, because then $p_{mr2d}=1, p_{mr1}=1$, and $p_{mr2}=1$, hence the first terms of the expressions for S2 and S3 are the entire expression for S1. Usually, however, p_{mr2d} will be close to 1 because an attacker will find it hard to believe if they are paying attention, and will be equal to 1 if the attacker is not paying attention, so S2 will be usually undesirable compared S1.

3.4 Multi-session reasoning

Thusfar we have assumed a single-session attack. However, an attacker could log in multiple times as in “low and slow” attacks. We can make a small change to the above model to handle this. Each pair of sessions can have an associated probability that they are due to the same user. If we trust our user authentication, this is 1 for sessions under the same name and 0 otherwise. However, attackers can masquerade as legitimate users, and this can be suspected when measured by anomaly-focused intrusion-detection systems.

Again we can use the exponential model. A user logging in again after time t after logging out under a different name is likely to be the same user with probability $p_S = p_S e^{-\lambda_{Sm} t}$ if they are malicious and zero if they are nonmalicious, where λ_{Sm} is the malicious-user "decay" rate and p_S is the probability that a malicious user logging out will immediately log in again in a unit time interval. If R users logged out recently, each with probability of maliciousness p_{mi} and at time t_i before the present, the probability a new user is now malicious is $p_{m0} + \sum_{i=1}^{R} p_S p_{mi} e^{-\lambda_{Sm} t_i}$ where p_{m0} is the a priori probability a new user would be malicious. But this is only an initial value; once the new session begins, this probability of being malicious will change with user actions as per the intrusion-detection system. Note we exclude users that are still logged in. Some of these could be conceivably opening separate login windows, but unless commands to the operating system are being generated by a software script, which can be detected by unusually fast commands, a malicious user cannot do more than one thing at a time.

For instance, suppose user “tom” logged in at 10PM and used the file system and the network connection to site “foobar” and transferred what looks to be a rootkit, logged out at 10:30PM, and then another username logged in at 10:35PM with no one else on the system. Suppose $\lambda_{Sm} = 0.1$ for times in minutes, $p_S = 0.4$, $p_{m1} = 0.8$, and $p_{m0} = 0.001$. The probability the new user is malicious is $0.001 + 0.4 * 0.8 * e^{-0.5} = 0.195$. Suppose the cost of a successful malicious attack is 20, the cost of hurting a legitimate user was 1, the probability of an attacker remaining after the particular deception "network not authorized" is 0.5, and the cost of a legitimate user remaining after this deception is 0.2. By our simple decision model, 0.195 > g((1*0.8)/(20*.5)) = g(0.08) = 0.08 and this deception is justified.

4. Consistency of deceptions

Repeated deceptions are a good way to foil many attacks when the defender is not at first sure of the nature of the attack. However, a key clue to deception that the attacker can recognize is inconsistent information (Vrij, 2000). Thus, besides applying the cost criteria of section 3, we should avoid deceptions that would logically contradict previously supplied information about the state of a computer system.

The resources associated with each possible command issued to an operating system are the "material" used in attacks (Templeton & Levitt, 2000). The main categories are:

- The directories and files of the computer;
- Peripheral devices to which the computer is attached;
- Networks to which the computer is attached;
- Other sites accessible by the networks;
- The executables for the commands run by an operating system;
- Status markers such as "logged-in" and "administrator privileges" (tokens that must be acquired);
- People associated with the computer systems. People are not always cooperative, although "social-engineering" attacks exploit them.

Resources can be identified as certain arguments to commands, as the site server23 for the command "ftp server23". Other resources are entailed by those already mentioned, like the “network” when transferring files with “ftp”. The executable associated with each command is also a resource, like the "ftp.bin" resource that implements the "ftp" command. It is also useful to distinguish the new resources created by commands, because resource-denial deceptions are more convincing on resources untested over time.

For each resource, we propose six facets of its status, each with an associated predicate:

- Existence, exists(X): Whether the resource X exists (in the place you are looking);
- Authorization, authorized(X): Whether the user is authorized to use the resource X (as by passwords and access control);
- Readiness, initialized(X,A): Whether the initialization of resource X is sufficient for the associated action A to be done;
- Operability, working(X): Whether the resource X is functionally sound;
- Compatibility, compatible(X,Y): Whether the two resources X and Y are mutually compatible (for instance, a text editor is incompatible with an image);
- Moderation, moderate(X,A): Whether the action's demands on a resource are within acceptable limits.

As an example consider the successful action by user Bob of downloading a file "foobar.doc" of size 50,000 bytes from "remotesite" to "homesite" across network "localnet" via the FTP file-transfer utility on homesite, at a time when localnet has five simultaneous users already. The resources necessary to accomplish this are file systems on remotesite and homesite, the network localnet, and the executables for ftp. Successful completion of the actions says that:

- File systems on remotesite and homesite exist.
- File systems on remotesite and homesite are authorized access by Bob.
- File systems on remotesite and homesite are initialized for access.
- File systems on remotesite and homesite are working.
- The network localnet exists.
- The network localnet is authorized use by Bob.
- The network localnet is initialized for file transfers.
- The network localnet is working.
- Localnet is compatible with remotesite and homesite.
- Executable ftp exists on homesite.
- Bob is authorized to use ftp on homesite.
- Executable ftp on homesite is initialized.
- Executable ftp on homesite is working.
- Executable ftp is compatible with the file system on homesite.
- Executable ftp is compatible with localnet.
- Executable ftp is compatible the file system on remotesite.
- The file system on homesite can hold files of 50,000 bytes.
- Localnet can transfer files of 50,000 bytes.
- Localnet can handle six simultaneous users.

Then any convincing deception we do after the download cannot violate any of these. For instance, if the user tries another download, we cannot now say that the download executable is missing, broken, or uninitialized.

5. Conclusions

We have examined a relatively new approach to defending computer systems, that of having them deliberately deceive attackers about resource availability to waste their time and eventually make them go away. Our models are straightforward to construct and can handle a wide range of situations. Because deception risks hurting legitimate users, we must carefully assess the costs and benefits, and our modeling will help us do this more precisely. Our models will be especially useful in automated responses to attacks because their precision permits quick use; automated responses are essential for critical information-system infrastructure such as military networks.

6. Acknowledgements

This work was supported by the U.S. National Science Foundation under the Cyber Trust Program. The views expressed are those of the author and do not represent policy of the U.S. Government.

7. References

Barber, R. S., & Kim. J. (2001) "Belief revision process based on trust: agents evaluating reputation of information sources," in Falcone, R., Singh, M., & Tan, Y.-H. (eds.), *Trust in Cyber-Societies*, LNAI 2246, Berlin: Springer-Verlag, pp. 73-82.

Chirillo, J. (2002) *Hack attacks revealed,* New York: Wiley.

Christian, D., & Young, R. C. (2004) "Strategic deception in agents," Proc. 3rd Intl. Joint Conference on Autonomous Agents and Multiagent Systems, New York, NY, pp. 218-226.

Cohen, F. (1999, May) "Simulating cyber attacks, defenses, and consequences," http://all.net/journal/ntb/simulate/simulate.html.

Cohen, F., & Koike, D. (2003) "Leading attackers through attack graphs with deceptions," *Computers and Security*, Vol. 22, no. 5, pp. 402-411.

DeRosis, F., Castelfranchi, C., Carofiglio, V., & Grassano, R. (2003) "Can computers deliberately deceive? A simulation tool and its application to Turing's imitation game," *Computational Intelligence*, Vol. 19, No. 3, pp. 235-263.

Ford, C. V. (1996) *Lies! Lies!! Lies!!! The Psychology of Deceit*, Washington, DC: American Psychiatric Press.

Gerwehr, S., Weissler, R., Medby, J. J., Anderson, R. H., & Rothenberg, J. (2000, November) "Employing deception in information systems to thwart adversary reconnaissance-phase activities," Project Memorandum, National Defense Research Institute, Rand Corp., PM-1124-NSA.

Heuer, R. J. (1982) "Cognitive factors in deception and counterdeception," in *Strategic Military Deception*, ed. Daniel, D. C., & Herbig, K. L., New York: Pergamon, pp. 31-69.

The Honeynet Project (2004) *Know Your Enemy, 2nd Edition,* Boston: Addison-Wesley.

Josang, A. (2001, June) "A logic for uncertain probabilities," *Int. Journal of Uncertainty, Fuzziness, and Knowledge-Based Systems*, Vol. 9, No. 3, pp. 279-311.

Korb, K., & Nicholson, A. (2004) *Bayesian artificial intelligence,* Boca Raton, FL: Chapman and Hall/CRC.

Lachmann, M., & Bergstrom, C. (2004) "The disadvantages of combinatorial communication," *Proc. of the Royal Society of London B*, Vol. 271, pp. 2337-2343.

Lydon, J., & Zanna, M. (1990, June) "Commitment in the face of adversity: A value-affirmation approach," *Journal of Personality & Social Psychology*, Vol. 58, No. 6, pp. 1040-1047.

Monteiro, V., (2003, March) "How intrusion detection can improve software decoy applications," M.S. thesis, U.S. Naval Postgraduate School, www.cs.nps.navy.mil/people/faculty/rowe/ oldstudents/monteiro_thesis.htm.

Nyberg, D. (1993) *The varnished truth: truth telling and deceiving in ordinary life*, Chicago: University of Chicago Press.

Rowe, N. (2004, December) "Designing good deceptions in defense of information systems," Computer *Security Applications Conference*, Tucson, AZ, pp. 418-427.

Rowe, N., & Rothstein, H. (2004, July) "Two taxonomies of deception for attacks on information systems," *Journal of Information Warfare*, Vol. 3, No. 2, pp. 27-39.

Rowe, N., Duong, B., & Custy, E. (2006, June) "Fake honeypots: a defensive tactic for cyberspace," 7th IEEE Workshop on Information Assurance, West Point, NY, pp. 223-230.

Snyder, C., Higgins, R., & Stucky, R. (1983) *Excuses: masquerades in search of grace,* New York: Wiley.

Templeton, S., & Levitt, K. (2000, September) "A requires/provides model for computer attacks," *Proc. of the New Security Paradigms Workshop*, Cork, Ireland.

Vrij, A. (2000) *Detecting lies and deceit: the psychology of lying and the implications for professional practice,* Chichester, UK: Wiley.

Whaley, B. (1982, March) "Towards a general theory of deception," *Journal of Strategic Studies*, Vol. 5, No. 1, pp. 179-193.

Zeller, A. (2002) "Isolating cause-effect chains from computer programs," *Proc. 10th ACM SIGSOFT Symposium on Foundations of Software Engineering*, Charleston, SC, pp. 1-10.

8. Appendix: Questionnaire and geometric mean of the responses

1a. What is the probability of finding a random computer's Internet connection down on a random day? .0416
1b. What is the probability of finding a random computer's local-area network down on a random day? .0422
1c. What is the probability that communications defaults are messed up so that binary file transfers across the Internet make the files unopenable? .0079
1d. What is the probability that a random computer will say that you are not an authorized user when you give it a correct password? .0057
1e. What is the probability that a random computer will deliberately deceive you? .0043
2a. What is the probability that a random system is wrong when it says the Internet is down? .0165
2b. What is the probability that a random system is wrong when it says its local-area network is down? .0086
2c. What is the probability that a random system will inadvertently mess up your files in binary transfers across the Internet? .0113
2d. What is the probability that a random system will lie to you when it says the Internet is down? .0072
2e. What is the probability that a random system will lie to you when it says its local-area network is down? .0032
2f. What is the probability that a random system will deliberately mess up your files in binary transfers across the Internet? .009
3a. Supposing you need to download a file as the major part of your job today, what is the probability you will log out immediately when the system says the Internet is down or it appears that the Internet is down? .0196
3b. Supposing you need to download a file as the major part of your job today, what is the probability you will log out immediately when you have tried three times to transfer files and the system says the Internet is down or it appears that the Internet is down each time? .1012
3c. Supposing you need to download a file as the major part of your job today, what is the probability you will log out immediately when the system says the local area network is down or it appears that the Internet is down? .0566
3d. Supposing you need to download a file as the major part of your job today, what is the probability you will log out immediately when you have tried three times to transfer files and the system says the local area network is down or it appears that the Internet is down each time? .0702
3e. Supposing you need to download a file as the major part of your job today, what is the probability you log out immediately when the files you transfer are messed up? .0052
3f. Supposing you need to download a file as the major part of your job today, what is the probability you log out immediately when you have tried three times to transfer files and the files you transfer are messed up each time? .0225