Teaching Context in Information Security

MATT BISHOP
University of California, Davis

This article investigates teaching the application of technical ideas by non-technical means, especially by using puzzles to engage students. After discussing the need to teach students to evaluate contexts in which decisions about computer security must be made, we suggest questions and scenarios drawn from political science, history, as well as other humanities, to force students to apply or derive principles of computer security in unusual and unexpected situations. Our experience shows that students find the process enjoyable, stimulating, and effective.

Categories and Subject Descriptors: D.4.6 **[Operating Systems]**: Security and Protection; K.3.2 **[Computers and Education]**: Computer and Information Science Education; K.4.0 **[Computers and Society]**: General; K.6.5 **[Management of Computing and Information Systems]**: Security and Protection

General Terms: Management, Security

Additional Key Words and Phrases: Environment, judgment, instruction, computer security

1. INTRODUCTION

Software engineers used standard software engineering processes to develop software. They installed the software in a state-of-the-art touch-screen system, and took great care to make the interface easy to use for even the most naïve user. They used encryption to protect the results being transmitted to a central server, powerful systems to provide the initialization and computation of results, and enabled an electronic audit log to validate the results. Then they applied an ISO 9000 process to ensure good quality control and had their product certified by both US federal and state testing laboratories. Last-minute patches and modifications were made to ensure that the systems worked. Yet when deployed and used for one day, the machines failed repeatedly [California Secretary of State 2004a]. As a result, the systems were decertified, and could not be used [California Secretary of State 2004b]. What went wrong?

The vendor sold electronic touch-screen voting systems (called DREs) and the server used in the clerk-recorders' offices to total the votes. The problems occurred because the vendor failed to recognize the particular context in which DREs are used. The public and the candidates expect voting systems to meet certain (sometimes nebulous) requirements, but three that all agree on are (1) correctness in recording the votes and in counting them (accuracy); (2) the guaranteed anonymity of the individual vote (privacy and secrecy); and (3) the easy availability of the DREs. Unlike standard systems, however, the DREs and the server are used only on one day, election day. Failures of most types of software are annoying, yet delays of an hour are not catastrophic. But in certain environments

This research was supported in part by National Science Foundation Grant 0311723 to the University of California, Davis.
Author's address: Matt Bishop, Department of Computer Science, University of California Davis, Davis, CA 95616-8562, bishop@cs.ucdavis.edu
A preliminary version of this paper was presented at the Sixth Workshop on Education in Computer Security in July 2004 [Bishop 2004].

and a government agency all differ. The problem is that the techniques that exist, and are (notably medical and aviation), delays may be catastrophic, and vendors use high-assurance techniques to ensure that chances of failure are minimal, and even then they usually provide backups to handle failure. If voting systems are not available, or inaccurate, voters are disenfranchised, which, in a democratic society, is unacceptable. The DRE vendor's error was in not realizing that the social context in which its product would be used resembled the time-critical environment more than the ordinary-use environment.

The vendor's error is a common one. Consider the popular phrase "securing cyberspace." The term "security" has too many meanings in this context. What are the requirements? They vary from industry to industry, indeed from organization to organization; the requirements for an academic institution, a company, a public charity, being developed, to secure systems are being deployed without adequate consideration of the context in which they will be used. "Are these methods appropriate and effective *for this particular environment*?" is a question heard all too infrequently.

A computer security course that provides a basic education in various aspects of securing computers and information must also teach students when to use techniques and, more importantly, how to analyze social, political, and cultural environments and contexts to determine whether a particular technique or technology is appropriate. This should not become all-consuming, to the detriment of teaching the technology and application of principles, but it must be attempted in some measure.

Doing so has several benefits. First, teaching the non-technical aspects treats security as a holistic problem rather than a purely technical one. This is reality. Mechanisms that are acceptable in some environments are not acceptable in others. For example, military environments can enforce procedural controls much more effectively than academic environments. In the former, the chain of command provides a framework for stating and enforcing procedures. In the latter, questioning and autonomy are encouraged to a much greater degree. Procedures must be created and promulgated differently to be effective.

But not only mechanisms differ; policies do, too. Counting votes in an organization is different than counting votes in a public election because the former can be rerun if there are irregularities. Worse, the policy may not be clear until *after* deployment, requiring both procedures and mechanisms to be adjusted accordingly.

Consider a state election in which voters punch holes in a paper ballot, and those ballots are counted using optical scanners. If the scanners are misaligned, the holes may not correspond to the correct votes and the scanners may misread the person or position being voted for. So the state requires that the scanners be checked before the votes are counted. It does so by requiring that the votes for 1% of the precincts be counted manually and then run through the optical scanners. Then the two vote totals are compared. If they disagree, the optical scanners must be recalibrated and the procedure repeated until the hand count and optical scan totals agree. Now suppose a new type of voting machine is used, which records all votes electronically with no paper record. Does generating paper copies of the votes from the electronic equipment, counting them, and then comparing those totals to the ones reported by the equipment satisfy the statutory 1% requirement? It does not if the intent is to verify that the voting machine records and totals the votes correctly, since the paper records are generated from the recorded votes, which are not independently checked against the choice the voter touched on the screen. It does if the intent is to verify that the voting machine totals the individual votes accurately. The usefulness of this recount is a matter of law, not technology.

No purely technical treatment of computer security can produce competent security experts. Policy defines security, and the policy is given *ex cathedra*. Policies need to account for people. They must counter human foibles and describe security in a particular context or set of contexts that includes environment, law, and expected user base. Hence security is a human issue, and teaching it as a purely technological one introduces gaps that technology cannot close.

The "best practices" rules gaining in popularity are good examples of this. Most describe how to secure a system. But many presume some definition of "secure" without stating it and without explaining *why* the choices are made. This can lead to contradictions within the rules. For example, one standard [CIS 2002] deals with (among other things) Windows 2000 Professional logs. In one place [CIS 2002, pp. 22-23], the rules prescribe that log events are to be overwritten as needed; in another [CIS 2002, p. 30], the rules prescribe that the system is to be shut down if the log fills to capacity.[1]

If full log files cause overwriting, shutdown, or disable logging, the "best practice" rules should prescribe a choice appropriate to the goals of the system and its environment. If the log space is large enough so that the log cannot be filled between snapshots (and the log is cleared after each snapshot), then the setting is irrelevant. Otherwise, the policy must indicate which choice will give the greater benefit (or do the least harm). People who have been exposed to the interaction of technical and non-technical matters will be able to make such judgments; those who have not will tend to follow the best practices without additional thought.

This judgment, the ability to apply principles and creativity in unusual or unexpected situations, is what computer security courses must encourage and teach. The key question is how to do so while keeping the students engaged and also supplement the technical aspects of the instruction.

2. SOLUTION

To meet these needs, a computer security course can tie security principles to experience and practice by drawing examples from other disciplines, which should focus on four general areas.

The first area is teaching students to question assumptions. This technique helps them to locate security flaws in a system by uncovering gaps among the security mechanisms; it also uncovers assumptions about the environment. Returning to the electronic voting machine example, one security audit downplayed the possibility of a voter inserting multiple cards to cast multiple votes by reasoning that the "voting machine makes a loud noise and ejects the smartcard after each vote is cast" [SAIC 2003, p. B-9]. This assumes that poll workers would hear a voting card being ejected, and thus be able to tell if a voter voted twice (because they would hear the second card being ejected too). But in practice, the noise at most precincts would mask that of the card being ejected [RABA Technologies 2004, p. 14]. This assumption is not valid in the particular environment in which the machines would be used.

This suggests that all parts of a problem must be examined, and seems counter to the traditional "top-down" methods of analysis customary in computer science. Those methods decompose problems into smaller ones, the smaller problems are solved, and the solutions combined to solve the larger problem. If each smaller problem is solved *in the context of the larger problem*, then the traditional approach works. All too often, though, the smaller solutions ignore the context of the larger problem and so do not work properly. A classic example comes from warfare, in which one assumes that winning all

[1] Later versions (for example, [CIS 2004]) corrected this inconsistency.

the battles means winning the war. King Pyrrhus would disagree. After he beat the Persians in one of the innumerable battles between the Greeks and Persians, his army was so decimated that he said, “One more victory like this, and we are undone.”

The second area, i.e., examining a problem as a whole, sometimes leads to unexpected solutions; it also offers the instructor the opportunity to emphasize the importance of looking at all aspects of a problem. A (possibly apocryphal) story of an attacker who repeatedly broke into computer systems makes this point. All sorts of technical measures were tried, but none succeeded. The defenders called the police, who tracked the perpetrator, a teenager, to another country that had no laws against this activity—so, naturally, when the police in that country were called, they declined to take action. The defenders were stymied until a policeman had an unusual idea. He called the teen-ager’s mother and told her what her son was doing. The attacks stopped immediately. The policeman’s observation that this was a human problem, and his thinking that a primal human emotion (love or respect for a parent) might solve the problem, was both astute and effective.

The third area emphasizes the need to consider human beings, both as individuals and as members of an organization. Security does not occur in a vacuum. Requiring users to authenticate themselves by providing urine specimens will not work in many societies because it violates customs of privacy; other, less invasive forms of biometrics such as fingerprint or retinal scans are more acceptable. Similarly, expecting low-level employees to refuse instructions from a corporate vice-president who can fire them is quixotic at best; the consequences to whistle-blowers can be severe [Glazer 2002]. Security measures must take these inhibitions into account.

An effective way to get students to understand these issues is to engage them by providing puzzles and asking the students to brainstorm creative solutions. This is the fourth area: think outside the box. It has two benefits: First, it forces the students to think and speak up, which allows the instructor to guide the discussion into considering a variety of approaches without providing an answer. (As we shall see, some puzzles simply have no correct answer.) Second, the students often enjoy a short break from technical material, and a good puzzle will lead them to either discover some principle, or apply some principle, that also applies to technical material. In our experience, the students enjoy working with puzzles, and sometimes suggest solutions or approaches, or complications, that the instructor had not considered; which also demonstrates the need for multiple viewpoints when considering security issues.

3. SELECTING PUZZLES

The two paramount rules for selecting a puzzle are that the puzzle must engage the students and must illustrate some principle relevant to computer security. The instructor must know something about the students to be able to engage them. If, for example, the students are from industry, the instructor should concoct puzzles from business organizations and the world of commercial information technology. At a university, puzzles that involve the administration are effective, as are puzzles drawn from other academic disciplines. For most students, current events can serve to supply puzzles, as can break-ins and responses to them. People also take serious interest in “war stories,” so puzzles that concern conflict, whether in war or politics, are very useful.

As for illustrating principles relevant to computer security, the instructor can use the relationship between the virtual “cyber” world and the physical “real” world. Principles of computer security derive from older principles. For example, the principle of least privilege [Saltzer and Schroeder 1975] is a variant of the “need-to-know” principle so

popular with governments and other organizations; and the principle of the separation of privilege [Saltzer and Schroeder 1975] is a formalization of the idea of "defense in depth". Other parallels abound. The instructor can take advantage of this to illustrate the relationships between the field of computer security and the constraints of human, organizational, environmental, and other concerns that students must consider.

Creating puzzles is straightforward, once one finds a topic. If the inspiration comes from a passage in a book or article, the instructor can ask the students how the passage demonstrates a specific principle. A more open-ended method is to ask the students which principles the passage demonstrates and why. A variant is to ask how to apply the contents of the passage to a computer system. This author prefers the open-ended approach for two reasons: First, if the discussion strays, the instructor can bring it back to the points he or she wishes to make; second, the students sometimes think of values or relationships that the instructor has not considered. The open-ended approach also encourages students to speak more freely, as the problem is not so constrained as in the first approach. In the author's experience, encouraging creativity requires persuading the students to express and consider ideas that sound crazy. Neils Bohr's comment that a colleague's idea was crazy, but not crazy enough to be true, is as apt in computer security as it is in quantum physics.

News stories and current events are also a fertile source for puzzles. Here the approach is slightly different. Rather than ask about general principles, the instructor can ask the students to put themselves in the position of the people involved in the incidents, or in the position of an analyst who seeks to prevent or enable the actions in the incident, and then say how they would act or what questions they would ask. The latter is particularly important. Students need to understand that security usually involves dealing with incomplete information, and part of what a good analyst does is ask questions. Taking this approach teaches the students how to analyze a problem, figure out what *additional* information would help them make decisions, and how to ask for it. When discussing fiction or historical incidents, this approach may fizzle: students may either feel too remote from the events, or they can "look in the back of the book" to see the answers; but they are witnesses to current events and can easily relate to them.

4. EXAMPLES

Some example puzzles in various areas of computer security follow. They were used in several undergraduate classes to spark thought and discussion among students, as well as to bring out points the instructor wished to emphasize. We presented one puzzle at the beginning of each class for the students to talk about among themselves for five minutes or so, and then discuss their conclusions and ideas. The puzzles fall into the four (broad) categories reviewed in Section 2.

4.1 Questioning Assumptions

Saul Alinsky's book *Rules for Radicals* [Alinsky 1972] provides fertile ground for the category of questioning assumptions. Alinsky took pride in organizing the poor and disenfranchised to force those in power to respond to their needs. His descriptions of tactics give insight into ways to attack systems, both political and computer.

The following is one of Alinsky's tactical rules for an organizer:

> "The third rule is: *Whenever possible go outside of the experience of the enemy*. Here you want to cause confusion, fear, and retreat. General William T. Sherman, whose name still causes a frenzied reaction throughout the South, provided a classic example of going outside the enemy's experience. Until Sherman, military tactics and strategies were based on standard

> patterns. All armies had fronts, rears, flanks, lines of communication, and lines of supply. Military campaigns were aimed at such standard objectives as rolling up the flanks of the enemy army or cutting the lines of supply or lines of communication, or moving around to attack from the rear. When Sherman cut loose on his famous March to the Sea, he had no front or rear lines of supplies or any other lines. He was on the loose and living on the land. The South, confronted with this new form of military invasion, reacted with confusion, panic, terror, and collapse. Sherman swept on to inevitable victory. It was the same tactic that, years later in the early days of World War II, the Nazi Panzer tank divisions emulated in their far-flung sweeps into enemy territory, as did our own General Patton with the American Third Armored Division." [Alinsky 1972, pp. 127-128]

Those who attack computers act as Sherman did: they ask about the assumptions the defenders are making and attack in ways that the defenders have not prepared for. Unless the defenders can handle situations they did not expect, they will react the way the South did: with confusion, panic, and collapse. Now consider security procedures for handling attacks. Applying the lesson from the passage above, students see that they must be prepared to depart from the standard procedures as needed, and know how and when to do so. Further discussion can educate the students to understand why in incident-handling rigidity will serve them ill, and flexibility well.

As another example, consider the following statement, offered as support of a Microsoft claim that Windows NT is more secure than Linux:

> Linux has not supported key security accreditation standards. Every member of the Windows NT family since Windows NT 3.5 has been evaluated at either a C2 level under the U.S. Government's evaluation process or at a C2-equivalent level under the British Government's ITSEC process. In contrast, no Linux products are listed on the U.S. Government's evaluated product list.

The underlying assumptions are that the security standards are meaningful and that they apply to the operating system. The first is true if the environments in which the systems are used match the environments for which the standards were developed. The second assumption is true only when the standards apply just to the operating system and do not depend upon the presence (or absence) of other software and hardware. Alas, most standards (and in particular the C2 standard cited in the passage) are based on the entire system, not only on the operating system.

4.2 Holistic Thinking

Holistic thinking asks students to look at problems in context rather than from a narrow technical perspective. The following puzzle illustrates one approach to encouraging this mode of thinking:

> Microsoft spent February of last year teaching its programmers how to check their code for security vulnerabilities and how to introduce common security flaws. Yet many Microsoft programs still have security vulnerabilities. What problems do you think Microsoft encountered, and will encounter, in trying to find and clean up the vulnerabilities in its systems?

Initially, students brainstorm the technical problems that Microsoft faces. But those are relatively minor compared to the multiplicity of environments in which Microsoft systems are used. Vulnerabilities are defined in terms of the local policy, and Microsoft cannot build systems to satisfy all those policies. So Microsoft programs will continue to

have vulnerabilities. Further, Microsoft supports backwards compatibility on their systems. Fixing vulnerabilities may break this feature. What are the trade-offs?

The instructor can also point out the difference between security vulnerabilities and poor coding practices. Buffer overflows may indicate security vulnerabilities; they always indicate non-robust coding problems [Bishop and Frincke 2004].

Another puzzle asks students to consider human nature as part of a problem:

> Some programs use passwords for access control, but do not protect the passwords in a very sophisticated manner (for example, by saving them in a file) or make determining the correct password very easy (for example, the Microsoft Word 5.0 encipherment scheme). The argument for using simple passwords and weak encipherment is that the data or programs being protected are of little value and the passwords give a small measure of privacy.

Given that what they are protecting is truly of little value, why is the use of such simple passwords and easily broken encipherments bad?

Again, students usually focus on the need to protect all data and the need to discourage attackers by making even worthless information difficult to acquire. While these needs are important, they overlook the problem of familiarity. Average users may forget about the weakness of the protection schemes and put sensitive data into the program. Further, people typically use the same password for many things. So if an attacker can compromise the password, he or she can then try the password in more sensitive environments, such as the login password—and will likely succeed.

4.3 Human and Organizational Problems

Sun Tzu's *The Art of War* [Sun Tzu 1983] and Niccolò Machiavelli's *The Prince* [Machiavelli 1995] are excellent sources for human and organizational problems, although many news stories provide fodder as well. Their point is that security must take people and organizations into account.

This passage from *The Prince* enables an instructor to illustrate how organizational problems affect security considerations:

> "It can be put like this: the prince who is more afraid of his own people than of foreign interference should build fortresses; but the prince who fears foreign interference more than his own people should forget about them. The castle of Milan, built by Francesco Sforza, has caused and will cause more uprisings against the House of Sforza than any other source of disturbance. So the best fortress that exists is to avoid being hated by the people. If you have fortresses and yet the people hate you they will not save you; once the people have taken up arms they will not lack for outside help. In our own time, there is no instance of a fortress proving its worth to any ruler, except in the case of the countess of Forli, after her consort, count Girolamo, had been killed. In her case the fortress gave her a refuge against the assault of the populace, where she could wait for succor from Milan and then recover the state. Circumstances were such that the people could not obtain support from outside. But subsequently fortresses proved of little worth even to her, when Cesare Borgia attacked her and then her hostile subjects joined forces with the invader. So then as before it would have been safer for her to have avoided the enmity of the people than to have had fortresses. So all things considered, I commend those who erect fortresses and those who do not; and I censure anyone who, putting his trust in fortresses, does not mind if he is hated by the people." [Machiavelli 1995, p. 69]

Technical courses rarely emphasize the importance of security officers obtaining and retaining the good will of the users and system administrators. Without that good will, the

officers will spend more time dealing with recalcitrant and upset authorized users than they will with attacks from outsiders. With that good will, many more people will report suspicious problems, and system administrators will be receptive to adding, configuring, and applying security mechanisms. The instructor can use this observation to lead into a discussion of organizational techniques and processes that will encourage this type of collaboration.

Sun Tzu illustrates a different aspect of organizational problems:

> "In A.D. 404, Liu Yu pursued the rebel Huan Hsuan up the Yangtze and fought a naval battle with him at the island of Ch'eng-hung. The loyal troops numbered only a few thousand, while their opponents were in great force. But Huan Hsuan, fearing the fate that was in store for him should he be overcome, had a light boat made fast to the side of his war junk, so that he might escape, if necessary, at a moment's notice. The natural result was that the fighting spirit of his soldiers was utterly quenched, and when the loyalists made an attack from windward with fireships, all striving with the utmost ardor to be first in the fray, Huan Hsuan's forces were routed, had to burn all their baggage, and fled for two days and nights without stopping." [Sun Tsu 1983, p.38]

This passage raises the question of special treatment. What happens when security rules apply to all *except* the managers? As Sun Tsu suggests, laxity trickles down from the top to all employees. Conversely, if the leaders of an organization respect the rules, employees are encouraged to do so as well. It is said that Alexander the Great was given a flask of water during his campaign in Persia; and upon being told his men had no water, he dumped the contents of the flask on the ground rather than have what his soldiers did not. Alexander was beloved by his men because he shared their hardships as well as their victories. This attitude carries over to security mechanisms and policies as well.

4.4. Thinking Out of the Box

Unusual problems demand creative solutions. The following story from *The Art of War* illustrates this point.

> "If we do not wish to fight, we can prevent the enemy from engaging us even though the lines of encampment be merely traced out on the ground. All we need to do is to throw something odd and unaccountable in his way.
>
> Tu Mu relates a strategm of Chu-ko Liang, who in 149 B.C., when occupying Yang-p'ing and about to be attacked by Ssu-ma I, suddenly struck his colors, stopping the beating of the drums, and flung open the city gates, showing only a few men engaged in sweeping and sprinkling the ground. This unexpected proceeding had the intended effect; for Ssu-Ma I, suspecting an ambush, actually drew off his army and retreated." [Sun Tsu 1983, pp.27–28]

Asking students to apply these stratagems to computer security can draw some interesting reactions. The key is to get students to think about how appearing to be weak, or unconcerned, may help improve security. After some discussion, this topic can lead to the role of deception in computer security, a very fertile area—and one, in this author's experience, that students enjoy.

Security personnel are given an assignment but, often, neither the power nor the information necessary to carry it out. The archetypal example is when one is told to "secure this company's network" but attempts to ascertain the needs of the organization are rebuffed. Handling this situation calls for creativity.

Isaac Asimov wrote a short story [Asimov 1984] that suggests a useful strategy for such a problem. In the story, a junior physicist discovers that he can levitate, but does not know how he does it. He needs help and support to study this phenomenon. Even though

he demonstrates this ability, people simply do not believe their own eyes; indeed, the chairman of his department says, "If I saw you fly, I'd see an optometrist or a psychiatrist. I'd sooner believe myself insane than that the laws of physics..." [Asimov 1984, p. 29]. The researcher writes letters to other physicists to ask for help, but is denounced as a crank. How can he get people to support his investigation?

The young physicist's solution is ingenious. He attends a major conference at which a highly respected physicist is to speak. During the talk he levitates, and when the speaker (who previously denounced him as a crank) challenges him, he denies he levitated and suggests that the speaker is crazy. This happens several times. The speaker calls government agents, has his own sanity tested, and finally confronts the levitator, who points out that anyone claiming to have seen someone levitate must be crazy because there is no explanation for such a phenomenon. Finally, the eminent physicist offers him research support if he will only levitate. The junior physicist promptly does so; thus helping the eminent physicist rather than the other way around. As one character says, "you let them let you help them."

Similarly, when security personnel are faced with a management that is unwilling to provide needed information, one approach is to talk directly to the users and find out what they mean when they ask for security. This is often quite different than what management *thinks* security should mean. It emphasizes the idea that co-operating with the end-users of the systems makes them feel that security is helping them. But this approach is unusual, in that it draws the needs of the users directly from the users themselves by involving the users in the process, thus sidestepping a potentially disastrous situation in which security interferes with work, resulting in, effectively, a denial of service attack.

5. CONCLUSION

Our experiences in using this technique have been uniformly good. In evaluations, students cite the "puzzle time" as a very enjoyable, educational aspect of the course. During the discussions, most of the students spoke up; the most serious problem we encountered was crowd control! The enthusiastic student participation in the discussions had the side benefit of encouraging them to ask questions during class. This approach makes clear to them that we encourage them to bring up ideas and misunderstandings, even when students think the ideas are stupid or silly. Sometimes, those "stupid or silly" ideas suggest approaches and insights that would otherwise be overlooked (and applies to both the non-technical aspects of the course as well as to the technical ones).

Judgment is a critical facility for engineers and scientists. But the trend in teaching computer science, and other engineering and science disciplines, is towards mathematical, analytical rigor. Analytic understanding and rigor are essential, but the ability to apply the techniques and technologies appropriately is equally important. Computer security is a science, but its application is also an art. The art of computer security needs to be emphasized far more than it is. Using puzzles drawn from non-engineering disciplines helps this process immensely.

APPENDIX: MORE EXERCISES

This appendix includes more examples.

Questioning Assumptions

As noted above, Saul Alinsky was a master at challenging common beliefs. He noted that societies often have sets of rules so complex that the consequences of following them to the letter are catastrophic. For example:

> "The basic tactic in warfare against the Haves is a mass political jujitsu: the Have-Nots do not rigidly oppose the Haves, but yield in such planned and skilled ways that the superior strength of the Haves become their own undoing. For example, since the Haves publicly pose as the custodians of responsibility, morality, law, and justice (which are frequently strangers to each other), they can be constantly pushed to live up to their own book of morality and regulations. No organization, including organized religion, can live up to the letter of its own book. You can club them to death with their "book" of rules and regulations. This is what the great revolutionary, Paul of Tarsus, knew when he wrote to the Corinthians: "Who also hath made us able ministers of the New Testament; not of the letter, but of the spirit; for the letter killeth.
>
> "Let us take, for example, the case of the civil rights demonstrations of 1963 in Birmingham, when thousands of Negro children stayed out of school to participate in the street demonstrations. The Birmingham Board of Education dusted off its book of regulations and threatened to expel all the children absent for that reason. Here the civil rights leaders erred (as they did on other vital tactics) by backing off instead of rushing in with more demonstrations and pressing the Birmingham Board of Education between the pages of their book of regulations by forcing them to live up to the letter of their regulations and statements. The Board and the City of Birmingham would have been in an impossible situation with every Negro child expelled and loose on the streets—if they didn't reverse themselves before they acted, they would have reversed themselves one day later." [Alinsky 1972, 152–153].

The assumption of the Board of Education was that the expulsion would punish the students. Alinsky points out that the students would indeed be punished—but the assumption that the expulsion would *only* affect the Negro population (the Have-Nots), and not the white population (the Haves) was wrong. The expelled students would have nothing to do during the day—and the potential for problems would increase accordingly.

Holistic Thinking

A fertile area for examples is the intersection of cryptography with system security. For example:

> A secure mailing system uses both the public key cipher RSA and the classical cipher DES. When one installs this system, the software generates two large (500 bits or so) numbers, to produce a modulus of 1024 bits. The private and public keys are generated from these quantities. The private key is enciphered with a classical cipher using a user-supplied pass phrase as the key. To send a message, a 64-bit key is randomly generated, and the message enciphered using DES with that key; the key is enciphered using the recipient's public key, and the message and enciphered key are sent.
>
> If you wanted to compromise a user's private key, what approaches would you take? Also, schemes like this are often said to give you the security of a 1024-bit key. Do you agree?

The first question leads to methods of tricking the user into revealing his or her key, or to finding the enciphered private key and trying to guess the pass-phrase, or inserting a keyboard recorder to obtain the pass-phrase directly. The point here is not to approach this as a problem in factoring; either social engineering or compromising the system will work better.

The second question is trickier, since we are tempted to assume the problem is asking about the discovery of *keys*. But it does not say that. It simply says "security." So students who think holistically will ask what "security" means here: Are we protecting the keys or the message? If the latter, guessing the random key used to envipher the message will suffice, even if that is simpler than determining the private key. The nature of the system and of the goals of the cryptosystem determine the answer to this puzzle.

Human and Organizational Problems

Sun Tzu gives an example of how an adversary can use his understanding of his opponent's character:

> "Yao Hsiang, when opposed in A.D. 357 by Huang Mei, Teng Ch'iang, and others, shut himself up behind his walls and refused to fight. Teng Ch'iang said: "Our adversary is of a choleric temper and easily provoked; let us make constant sallies and break down his walls, then he will grow angry and come out. Once we can bring his force to battle, it is doomed to be our prey." This plan was acted upon, Yao Hsiang came out to fight, was lured on as far as Sanyuan by the enemy's pretended flight, and finally attacked and slain." [Sun Tzu 1983, p. 40]

By shaping his tactics to lure an enemy into a place where it could be attacked, Teng Ch'iang had to know enough about his enemy to understand his weakness: a choleric temper, easily provoked. Had Teng Ch'iang not understood the enemy's character, the enemy would have stayed behind his walls, and the attackers would have had to endure a long siege. But Teng Ch'iang used the enemy's human weakness to create a situation in which he could use it to overcome his enemy.

Thinking Out of the Box

An apocryphal story shows students the importance of thinking "out of the box."

> During a six-month period, a number of computer installations were attacked by an intruder who broke in and simply looked at the data on the system. After repeated investigations, it was determined the intruders were from the Netherlands. The Dutch police were asked to investigate, because one of the computers was at a military site, and there was considerable belief that espionage against the United States was being committed.
>
> After a thorough investigation, the Dutch authorities found that the intruder was a high-school student who had no previous record of trouble. The authorities determined that he was not spying but simply amusing himself. They declined to proceed any further as attacking computer systems was not (then) a crime under Dutch law.
>
> The intruder continued to break into these systems despite efforts to stop him. While he caused no damage, he tied up lots of the system programmers' time. Someone finally suggested a way to stop the attacker from returning. The defenders implemented it and the problem ended.

That is the puzzle, but what is the solution? Students usually focus on technical methods to solve problems. Some of the more creative students ask about ways to use the law or site procedures to stop the attacker. These students are going beyond the technical, into the realm where the solution lies —the security officers called the high-school student's mother, explained what her son was doing, and asked her to stop him, which she did.

REFERENCES

ALINSKY, S. 1972. *Rules for Radicals*. Random House, New York.

ASIMOV, I. 1984. Belief. In *The Winds of Change*, Ballantine Books, New York, 12-46.

BISHOP, M. 2004. Teaching context in information security. In *Avoiding Fear, Uncertainty, and Doubt Through Effective Computer Security Education: Proceedings of the Sixth Workshop on Education in Computer Security* (Monterey, CA, July), Center for Information Systems Security, Naval Postgraduate School, Monterey, CA, 29-36.

BISHOP, M., AND FRINCKE, D. 2004. Teaching robust programming. *IEEE Security & Privacy 2*, 54-57.

CALIFORNIA SECRETARY OF STATE. 2004a. *Report on March 2, 2004 Statewide Primary Election*, April 20, 2004. Office of the Secretary of State, State of California, Sacramento, CA.

CALIFORNIA SECRETARY OF STATE. 2004b. *Decertification and Withdrawal of Approval of AccuVote-TSx Voting System as Conditionally Approved November 20, 2003, and Recission of Conditional Approval*, April 30, 2004. Office of the Secretary of State, State of California, Sacramento, CA.

CIS (CENTER FOR INTERNET SECURITY). 2002. *Professional Operating System Benchmark Consensus Baseline Security Settings, Version 1.0*. Center for Internet Security. http://www.cisecurity.org/.

CIS (CENTER FOR INTERNET SECURITY). 2004. *Windows 2000 Professional Operating System Level 2 Benchmark Consensus Baseline Security Settings, Version 2.2.1*. Center for Internet Security http://www.cisecurity.org/bench_win2000.html.

GLAZER, M.P. 2002. Ten whistleblowers: What they did and how they fared. In *Corporate and Governmental Deviance: Problems of Organizational Behavior in Contemporary Society*, 6th ed. Oxford University Press, New York, 229–249.

MACHIAVELLI, N. (1995 translation). *The Prince*. Penguin Books, New York.

RABA TECHNOLOGIES, INC. 2004. *Trusted Agent Report: Diebold AccuVote-TS Voting System*. RABA Technologies, Columbia, MD.

SAIC. 2003. *Risk Assessment Report: Diebold AccuVote-TS Voting System and Processes, Appendix B: Security Statements from the Rubin Report & State of Maryland Controls*. SAIC-6099-2003-261. http://www.verifiedvoting.org/downloads/votingsystemreportfinal.pdf.

SALTZER, J., AND SCHROEDER, M. 1975. The protection of information in a computer system. *Proc.IEEE 63*, 9, 1278–1308.

SUN TZU. (1983 translation). *The Art of War*. Dell Publishing, New York.

Received March 2005; accepted December 2005