

The Devil You Know: Responding to Interface-based Insider Attacks

Ronald L. Mendell 2002-02-06

The Devil You Know: Responding to Interface-based Insider Attacks

by *Ronald L. Mendell*

last updated February 6, 2002

Carl made a mistake. In his repetitious data entry job he entered employee information every workday. He always was careful to input the correct job requisition number in the user screen's JRN field. "Without a correct JRN entered, the new employee input won't process," his supervisor told him the first day. This time instead of "34896KN" his fingers danced the wrong way with an input of "34896KL." The input processed. Carl was able to go into the EMP_DATA file and correct it. The procedure was a bit of a pain, but he learned a valuable lesson his employer never meant for him to know. He realized he could set up bogus new employees on the payroll using a dummy JRN. By entering the wrong input he won the jackpot - his employer lost big time.

It is estimated that up to eighty-five percent of attacks are perpetrated by insiders (for example, in [Internet Wake-up Call: Are Financial Institutions Ready for Cyber Terrorists](#), by MacDonnel Ulsch published by Price Waterhouse Cooper Technology Risk Services). These attackers often stumble across weaknesses in the user interface design, and hijack them as vectors for system compromise. As these attackers are already on the network, they do not have to circumvent peripheral network defences. Instead, they are able to use their computers, computers they are fully authorized to use, to gain illicit access to information. Many of these attacks take advantage of problems that are inherent in user interfaces.

Since interfaces seek to be user-friendly, they often make the intruder's job easier. Investigators need to understand the vectors for attack interfaces offer. The article makes suggestions for investigators and response teams to detect and investigate interface-based insider attacks. It is also hoped that this article will provide the basis of incident response policies for responding to and investigating insider attacks that exploit interface-based vulnerabilities.

Cognitive Psychology

Successful human communication relies heavily on feedback. People require a response not

only from their fellow humans but from machines as well. So, to increase user performance, many computer interfaces tell the user when they have done something right ("Processed Correctly"), done something wrong ("File Access Denied"), or entered something formatted incorrectly ("Password must be five characters"). While most users appreciate these signposts, they are double-edged, providing potential aid and assistance to hostile users.

Cognitive psychology is concerned with how people acquire and process information. As such, it can provide an understanding of the interaction between humans and computers. Donald A. Norman in *The Design of Everyday Things* identifies three stages in a user's evaluation of a system:

1. Perceiving the state of the world;
2. Interpreting the state of the world; and,
3. Evaluating the outcome.

In other words, the user pays attention to what the machine actually does, not what the designer intended the machine to do. Users also tend to see what they are looking for. In the case of inside attackers, this means interface-based weaknesses.

The users' interface design may make a criminal's job easy or difficult. "Insignificant" details in the interface's behavior may become opportunities for exploitation by the savvy criminal. Attackers often see opportunity when the system behaves in unexpected or unintended ways. For example, a medical insurance processing program will not issue a check for greater than \$2,500.00 without special approval on a maternity claim. But, if the user enters "M" instead of "F" for the patient's sex, the program will issue checks for greater than \$2,500. This may be bizarre behavior for the system, something the designer did not anticipate; but if a user is hoping to rip-off the program, this is exactly the sort of inconsistency he or she will deliberately try to find. Manipulating other fields like age, city, state, a date, and so on may have similar effect. The users' interface design may make a criminal's job easy or difficult. "Insignificant" details in the interface's behavior become opportunities for exploitation by the savvy criminal. Savvy investigator needs to pick up on these idiosyncratic system behaviors when investigating computer crimes.

With this information in mind, it becomes readily apparent that, in investigating incidents such as these, forensic personnel should look for obviously anomalous information being entered into programs. In the example cited above, an applicant for maternity claims who is listed as a male

should stick out like a sore thumb. Any sort of anomalous information like this should be investigated. Furthermore, in implementing software programs, it may be helpful to have IT staff deliberately try to 'break' the interface by entering obviously incorrect information, just to see how the program reacts.

Obviously, not all users who discover unusual system behavior gravitate to computer crime. Psychologist Dr. George F. Nixon, who was interviewed for this article, observed: "the individuals involved in computer crime tend to be isolated socially. And, their isolation makes them sensitive and attuned to every nuance of the machine." In other words, the computer screen becomes a persona, one that will reveal the information that a keen audience can utilize. Any information that the system reveals may expose potential weaknesses in the system, which will only encourage additional probing. As Donald Norman comments, "People rely on what's in their head rather than what the system says." Security that underestimates the user's intelligence only fosters an attacker's resolve to prove his or her superior gamesmanship.

Underestimating the User's Intelligence

In the attempt to make user interfaces easier to use, security design can be an afterthought. It often gets added on after the completion of the overall interface. And it is usually geared toward people of average intelligence, people who will not challenge the interface's behavior. Hackers, on the other hand, characteristically think outside of the box. They may not possess extraordinary intelligence, but they do question appearances. They are ravenous for clues about what's happening behind the interface.

If a system's security measures have low complexity - what Bruce Schneier calls "low entropy" - patterns will emerge that attackers will pick up on. On the other hand, the interface may contain technical information that may be meaningless to the average user but not to an attacker. Investigators need to recognize that what the average user would have done or could have done with the interface doesn't count for much. Rather, investigators need to understand the way attackers think, so that they can anticipate what the hacker sees when he or she is working with the interface.

Feeble interface security only inhibits the timid. Hackers, on the other hand, are defined by their persistence. As a result, weaknesses that may not attract the notice of the average user serve as a puzzle to solve for attackers. As such, rather than serving as a deterrent, poor security design may encourage attackers. Examples of feeble security design include:

- **Weak password generation**, such as passwords that have low complexity or are easy to guess. Password patterns become highly predictable; for example, every tenth password created ends in "1045" and every ninth ends with "945" and so on.
- **Insufficient data constraints** according to which constraints are either nonexistent or not applied uniformly. This allows anomalous data to be added to the system. Borrowing from an earlier example, it may not be possible to designate males as eligible for maternity leave, but it may be possible to classify file clerks in the executive salary if adequate data constraints are not in place.
- **Inadequate cloaking of data**, such as if a Web site or input screen contains extraneous data that is not necessary for the conduct of business. This information may include: (a) Internal e-mail addresses, (b) server names or IP addresses, (c) internal telephone numbers and the names of company staff, (d) file names, directory structures, or data paths, and (e) feedback on what data ranges are acceptable for processing.

Systems That Disclose Too Much Information

As indicated in the preceding discussion of cloaking issues, some systems disclose unnecessary information to users. For instance, at log-in, the system may reveal the version of the OS that it is running on, the version of software that is running on the system, previous log-in information, the length and format of acceptable passwords, and which other users are logged in. Often, these obvious clues to the system's inner workings are part of the user interface's default configuration. With a little effort, administrators can hide this information by avoiding default installations in graphic or text-based interfaces. This can be important in case of an investigation: when investigating systems penetrations that pass through the user interface, always enquire on whether the administrator's configuration was ever changed. If the system defaults have been changed, this may provide a valuable clue in the investigation.

Investigators have to be adept at interpreting intruders' behavior. Every intrusion on a system is a crime scene. Reading that scene correctly helps to build an electronic trail back to the penetration point and then to the attack's point of origin. If the intruder directly uses an OS or software exploit without prior system probing then suspect they may have gained valuable information at the interface. Using the correct password format early in an attack is another clue that points to the interface as a guidepost. And, quickly heading for specific user's accounts or files reveal the attacker may have obtained insight from user status data available on the interface.

When investigating a computer crime that originated at a user interface, trying to play with various inputs may reveal useful information. Do the unexpected and see what results occur. Force unusual data combinations and previously unforeseen formats for fields. Creativity is key to investigating this crime of opportunity. A checklist of input testing would include:

1. Entering out-of-range data entry, such as the age of ten years old for an adult;
2. Entering conflicting data into two or more fields;
3. Entering inconsistent dates;
4. Leaving out data in "required fields",
5. Moving files to different folders or directories in a graphical tree and see if that changes the file's access restrictions; and,
6. Testing other input devices such as bar code readers to make they are functioning correctly.

If the user interface was developed in-house, investigators should consult the developers to find out what might have happened. If the interface is a commercial software product, do the standard intelligence gathering using search engines, checking news groups, and networking in the security community to find out what others have experienced in system compromise.

Cumbersome Security

In investigating the incident, the response team may realize that the network logs point to Ms. 'X' accessing a restricted database on 1-9-02 at 8:38 a.m. from her office PC. Unfortunately for the inquiry, she was on company business in Australia on that date. How could this have happened, and what are the ramifications for the investigation?

Opportunities for masquerading may be facilitated by having too many security requirements in place. Too much security creates problems for users and for security management. Needlessly complicated security logins or procedures on the user interface screen can quickly become a nuisance. Log-in steps of more than two passwords present a hurdle users tire of having to do repeatedly. Logins composed of several steps may tempt users to leave their PCs or terminals on constantly in an unlocked state. This opens up their terminals to piggybacking by unauthorized users.

Lengthy delays in issuing passwords to users or the complexity of changing passwords may encourage users to seek unofficial avenues for access. They may borrow other users' log-in

credentials, which creates problem in tracking who enters the system. Furthermore, such mundane social engineering strategies as shoulder surfing or dumpster diving may allow inside attackers to gain access to other employees' systems to perpetrate their attacks.

When an audit trail or a log in an investigation leads investigators to a contradiction, they should consider how someone might have been to masquerade as Ms. 'X'. Investigators should interview the victim. Questioning may produce the identities of those workers who "borrowed" logins for "emergencies" or those employees who knew where the target hid the sticky notes with passwords written on them.

The System Provides Clues on Cheating

Often, if the interface is used as the attack vector, the intruder will have tried to gain access multiple times, often by trying to guess or brute-force passwords. Investigators need to understand how multiple guessing can be used by to overcome interface password protection. For instance, if investigators see a sudden rise in user password attempts and then an accompanying fall off just prior to an incident, they can justifiably suspect a penetration at the interface. Such evidence may help the forensics team to develop an electronic trail to solve the case.

Access Rules Not Enforced

Another more subtle design problem involves directory/folder tree available in many applications. Does changing a file's folder alter the access privileges to the object? If the access rights depend upon the file's affiliation to the folder, then a savvy user may be able to access a sensitive file by changing the file's folder. For example the file EXECESALARY.xls is in the folder RESTRICTED. Only a user with rights to RESTRICTED can read EXESALARY.xls. But, what happens in a copy command that moves a copy of the file to folder PRYINGEYES? If the object is correctly restricted, it will retain the proper access rights and protection. Testing through the manipulating of file locations can prevent breaches of security like the file folder-copying scam.

When an investigator looks into the compromise of a sensitive file, he or she should check to see if it was available through a directory tree interface. Finding the file's copy in folders not normally associated with the original file indicate a compromise. File modification or creation dates and times will narrow the time frame on who might have had access.

Developing the Incident Response Policy

Web pages conveying security status information are becoming the norm. The pages serve as a monitoring tool to alert staff of deviations from the security status quo. Making sure that the security staff can spot a warning of an intrusion rapidly should be a priority. Security Web pages generally monitor the number of users on the system, provide output for the IDS, and offer a readout from "traps" – out of range system behavior. Server status is also another monitored area. Activity usually follows a color-coded system. Green backgrounds indicate within acceptable ranges. Yellow means borderline activity. Red indicates system failure or a security breach.

The incident response policy should contain clear written documentation telling staff what to do with different alerts. Carefully reviewing this documentation should be part of any investigation. If instructions are not clear, security staff may not respond to an alert correctly.

Any investigation of computer security violations has two purposes: (1) identify those responsible and (2) prevent the incident from happening again. If your network's security plan includes a NOC (Network Operations Center), make sure that any Web pages you create provide clear, unambiguous alerts to the NOC staff of intrusions, improper data entry, or manipulations and unauthorized entry to sensitive files and data bases. Key alert icons or windows must communicate unambiguous signals such as a flashing color background when something is wrong. Even if minimized, these warning icons must produce an eye-catching signal when something is amiss.

An investigator looking into an intrusion or security violation should interview the workers carefully. Do they understand how to interpret warnings and alerts? Do they know whom to notify? If documentation on the time of an attack is ambiguous, they may be able to identify when the look of pages changed. In addition, they may be able to correlate non-computer security page data with security events. For example, if a rise in proxy server activity accompanied the hack or if increased access to the computer lab shows on the physical access log page just before an attack began.

Always recognize the danger of human fatigue on the investigator's part in working the case. Often an investigator will have to go through voluminous log files during an inquiry. This is tedious and error-prone work. Whenever feasible use text searching tools available to reduce the process as much as possible. (See the References section below.)

Once the mechanism of the crime becomes clear, associate people to the crime by checking of the following:

- Access logs: Who and when people logged onto the system.
- VPN logs: Who accessed the system via the Virtual Private Network.
- Work Schedules: They establish when employees had access to hardware or software.
- Project Reports: Documents when employees worked with the software.
- Sign-in logs: Documents when employees were on the premises.
- Firewall/Proxy logs: Who and when users entered the network from the Internet via which services.
- Network logs: Who, What, Where, When, Why, and How.
- Physical surveillance: Videotape, physical access records of people using magnetic badges.
- Web server logs: Who has been hitting your site?
- Traffic logs on the interface: The amount of activity and when. Who accessed the interface?

Conclusions

Many organizations do not want to acknowledge that insider attacks are a real threat. However, statistics indicate that they are, in fact, the most common threat. As such, it is crucial that organizations, and their incident response teams, be prepared to detect and investigate insider attacks. As many insider attacks will be launched via an authorized system interface, understanding interface-based attacks is also crucial to the successful investigation of an insider attack. It is hoped that this article will serve as a valuable training resource for investigators who must prepare for the investigation of interface-based insider attacks. For more detailed information, please refer to the resources cited below.

References

Norman, Donald A., *The Design of Everyday Things*, Doubleday, 1988: The book serves as a good introduction to design engineering and to cognitive psychology.

Knightmare, The, *Secrets of a Super Hacker*, Loompanics, 1994: A basic, introductory hacker text, but it has some disturbing suggestions on gaining control of public terminals and PCs. It definitely identifies the little clues a hacker looks for in a user interface.

Schiffman, Mike, lead author/editor, *Hacker's Challenge*, Osborne/McGraw Hill, 2001: A must read book for computer security professionals for its creativity in recognizing the wide range of system attacks. If a common theme dominates this book, then it is that default configurations on servers, firewalls, and user interfaces cause most computer security ills. The book presents examples of a wide range of computer logs used in computer crime investigations and suggests tools for filtering the voluminous data the logs create. Investigating attacks on interfaces requires synthesis of diverse computer and non-computer information.

Tools for examining logs: (1.) AWK, a text searching language (for UNIX logs), (2.) [Samspace](#), a Web site for tools that aid in researching IP addresses, hosts, and domains, (3.) "[A Small Guide to Text Editors](#)", and (4.) [Pslist](#), it lets you list system processes (like in UNIX) in a Windows environment.

Dougherty, Dale and Robbins, Arnold, *Sed & AWK*, O'Reilly, 1997: A good introduction to the AWK language.

Mendell, Ronald L., [Vulnerable Databases: Sleuthing with AWK and SQL](#): An overview of using AWK and SQL in analyzing logs.

[Privacy Statement](#)

Copyright 2006, SecurityFocus