The Use of Deception Techniques: Honeypots and Decoys

Fred Cohen

Key words: Honeypots, Honenets, Deception, Human engineering, Perception management, Decoys, Cognitive deception

Abstract: Honeypots and similar sorts of decoys represent only the most rudimentary uses of deception in protection of information systems. But because of their relative popularity and cultural interest, they have gained substantial attention in the research and commercial communities. In this paper we will introduce honeypots and similar sorts of decoys, discuss their historical use in defense of information systems, and describe some of their uses today. We will then go into a bit of the theory behind deceptions, discuss their limitations, and put them in the greater context of information protection.

1. Background and History

Honeypots and other sorts of decoys are systems or components intended to cause malicious actors to attack the wrong targets. Along the way, they produce potentially useful information for defenders.

1.1 Deception fundamentals

According to the American Heritage Dictionary of the English Language (1981):

"deception" is defined as "the act of deceit"
"deceit" is defined as "deception".

Fundamentally, deception is about exploiting errors in cognitive systems for advantage. History shows that deception is achieved by systematically inducing and suppressing signals entering the target cognitive system. There have been many approaches to the identification of cognitive errors and methods for their exploitation, and some of these will be explored here. For more thorough

coverage, see [68]. Honeypots and decoys achieve this by presenting targets that appear to be useful targets for attackers. To quote Jesus Torres, who worked on honeypots as part of his graduate degree at the Naval Postgraduate School:

"For a honeypot to work, it needs to have some honey"

Honeypots work by providing something that appears to be desirable to the attacker. The attacker, in searching for the honey of interest, comes across the honeypot, and starts to taste of its wares. If they are appealing enough, the attacker spends significant time and effort getting at the honey provided. If the attacker has finite resources, the time spent going after the honeypot is time not spent going after other things the honeypot is intended to protect. If the attacker uses tools and techniques in attacking the honeypot, some aspects of those tools and techniques are revealed to the defender in the attack on the honeypot.

Decoys, like the chaff used to cause information systems used in missiles to go after the wrong objective, induce some signals into the cognitive system of their target (the missile) that, if successful, causes the missile to go after the chaff instead of their real objective. While some readers might be confused for a moment about the relevance of military operations to normal civilian use of deceptions, this example is particularly useful because it shows how information systems are used to deceive other information systems and it is an example in which only the induction of signals is applied. Of course in tactical situations, the real object of the missile attack may also take other actions to suppress its own signals, and this makes the analogy even better suited for this use. Honeypots and decoys only induce signals, they do not suppress them. While other deceptions that suppress signals may be used in concert with honeypots and decoys, the remainder of this paper will focus on signal induction as a deceptive technique and shy away from signal suppression and combinations of signal suppression and induction.

1.2 Historical Deceptions

Since long before 800 B.C. when Sun Tzu wrote "The Art of War" [28] deception has been key to success in warfare. Similarly, information protection as a field of study has been around for at least 4,000 years [41]. And long before humans documented the use of deceptions, even before humans existed, deception was common in nature. Just as baboons beat their chests, so did early humans, and of course who has not seen the films of Khrushchev at the United Nations beating his shoe on the table and stating "We will bury you!". While this article is about deceptions involving computer systems, understanding cognitive issues in deception is fundamental to understanding any deception.

1.3 Cognitive Deception Background

Many authors have examined facets of deception from both an experiential and cognitive perspective. Chuck Whitlock has built a large part of his career on identifying and demonstrating these sorts of deceptions. [12] His book includes detailed descriptions and examples of scores of common street deceptions. Fay Faron points out that most such confidence efforts are carried as as specific 'plays' and details the anatomy of a 'con' [30]. Bob Fellows [13] takes a detailed approach to how 'magic' and similar techniques exploit human fallibility and cognitive limits to deceive people. Thomas Gilovich [14] provides in-depth analysis of human reasoning fallibility by presenting evidence from psychological studies that demonstrate a number of human reasoning mechanisms resulting in erroneous conclusions. Charles K. West [32] describes the steps in psychological and social distortion of information and provides detailed support for cognitive limits leading to deception.

Al Seckel [15] provides about 100 excellent examples of various optical illusions, many of which work regardless of the knowledge of the observer, and some of which are defeated after the observer sees them only once. Donald D. Hoffman [36] expands this into a detailed examination of visual intelligence and how the brain processes visual information. It is particularly noteworthy that the visual cortex consumes a great deal of the total human brain space and that it has a great deal of effect on cognition. Deutsch [47] provides a series of demonstrations of interpretation and misinterpretation of audio information.

First Karrass [33] then Cialdini [34] have provided excellent summaries of negotiation strategies and the use of influence to gain advantage. Both also explain how to defend against influence tactics. Cialdini [34] provides a simple structure for influence and asserts that much of the effect of influence techniques is built-in and occurs below the conscious level for most people. Robertson and Powers [31] have worked out a more detailed low-level theoretical model of cognition based on "Perceptual Control Theory" (PCT), but extensions to higher levels of cognition have been highly speculative to date. They define a set of levels of cognition in terms of their order in the control system, but beyond the lowest few levels they have inadequate basis for asserting that these are orders of complexity in the classic control theoretical sense. Their higher level analysis results have also not been shown to be realistic representations of human behaviors.

David Lambert [2] provides an extensive collection of examples of deceptions and deceptive techniques mapped into a cognitive model intended for modeling deception in military situations. These are categorized into cognitive levels in Lambert's cognitive model. Charles Handy [37] discusses organizational structures and behaviors and the roles of power and influence within organizations. The National Research Council (NRC) [38] discusses models of human and organizational behavior and how automation has been applied in this area. The NRC report includes scores of examples of modeling techniques and details of simulation implementations based on those models and their applicability to current and future needs. Greene [46] describes the 48 laws of power and, along the way, demonstrates 48 methods that exert compliance forces in an organization. These can be traced to cognitive influences and mapped out using models like Lambert's, Cialdini's, and the one we describe later in this paper.

Closely related to the subject of deception is the work done by the CIA on the MKULTRA project. [52] A good summary of some of the pre-1990 results on psychological aspects of self-deception is provided in Heuer's CIA book on the psychology of intelligence analysis. [49] Heuer goes one step further in trying to start assessing ways to counter deception, and concludes that intelligence analysts can make improvements in their presentation and analysis process. Several other papers on deception detection have been written and substantially summarized in Vrij's book on the subject.[50]

All of these books and papers are summarized in more detail in “A Framework for Deception” [68] which provides much of the basis for the historical issues in this paper as well as other related issues in deception not limited to honeypots, decoys, and signal induction deceptions. In addition, most of the computer deception background presented next is derived from this paper.

1.4 Computer Deception Background

The most common example of a computer security mechanism based on deception is the response to attempted logins on most modern computer systems. When a user first attempts to access a system, they are asked for a user identification (UID) and password. Regardless of whether the cause of a failed access attempt was the result of a non-existent UID or an invalid password for that UID, a failed attempt is met with the same message. In text-based access methods, the UID is typically requested first and, even if no such UID exists in the system, a password is requested. Clearly, in such systems, the computer can identify that no such UID exists without asking for a password. And yet these systems intentionally suppress the information that no such UID exist and induce a message designed to indicate that the UID does exist. In earlier systems where this was not done, attackers exploited the result so as to gain additional information about which UIDs were on the system and this dramatically reduced their difficulty in attack. This is a very widely accepted practice, and when presented as a deception, many people who otherwise object to deceptions in computer systems indicate that this somehow doesn’t count as a deception.

1.4.1 Long-used Computer Deceptions

Examples of deception-based information system defenses that have been in use for a long time include concealed services, encryption, feeding false information, hard-to-guess passwords, isolated sub-file-system areas, low building profile, noise injection, path diversity, perception management, rerouting attacks, retaining confidentiality of security status information, spread

spectrum, steganography, and traps. In addition, it appears that criminals seek certainty in their attacks on computer systems and increased uncertainty caused by deceptions may have a deterrent effect. [40]

1.4.2 Honeypots

In the early 1990s, the use of honeypots and decoys as a deception in defense of information systems came to the forefront with a paper about a "Jail" created in 1991 by AT&T researchers in real-time to track an attacker and observe their actions. [39] An approach to using deceptions for defense by customizing every system to defeat automated attacks was published in 1992, [22] while in 1996, descriptions of Internet Lightning Rods were given [21] and an example of the use of perception management to counter perception management in the information infrastructure was given [23]. More thorough coverage of this history was covered in a 1999 paper on the subject. [6] Since that time, deception has increasingly been explored as a key technology area for innovation in information protection.

1.4.3 Deception ToolKit, D-WALL, Invisible Router, Responder, and Execution Wrappers

The public release of the Deception ToolKit (DTK) [19] led to a series of follow-on studies, technologies, and increasing adoption of technical deceptions for defense of information systems. This includes the creation of a small but growing industry with several commercial deception products, HoneyD from the HoneyNet project, the RIDLR project at Naval Post Graduate School, NSA-sponsored studies at RAND, the D-Wall technology, [66] [7], the Invisible Router, Responder [69],and a number of studies and commercial developments now underway. Deception toolkit was made available on a bootable Linux CD in the late 1990s as part of the White Glove Linux distribution. HoneyD is also not provided on a bootable CD from the HoneyNet project.

DTK creates sets of fictitious services using Perl and a deception-specific finite state-machine specification language to implement input, state, and output sequences that emulate legitimate services to a desired level of depth and fidelity. While any system can be emulated with this technology at the application layer, in practice the complexity of finite state machines is fairly limited. On the other hand, by the time the attacker is able to differentiate legitimate from DTK services, DTK has already alerted response processes and, with automated responses, other real services can be turned to deceptions to counter further attacks. Low-level deceptions that emulate operating systems at the protocol level are implemented in the White Glove version of DTK by setting kernel parameters using the /proc file system to emulate time to live (TTL) and other fields to increase the fidelity of the deception, however these effects are somewhat limited.

DWALL uses multiple address translation to allow a small number of computers to behave as if they were a larger number of computers. In the DWALL approach to deception, a large address space is covered by a small set of computers of different types that are selectively applied to different applications depending on the addresses and other control factors. DWALL provides the means for translating and selectively allowing services to be invoked so that each physical machine used as a high fidelity deception can be applied to a large number of addresses and appear to be a variety of different configurations. The translation is done by the DWALL while the high fidelity deception is done by a computer of the same type as the computer being projected to the attacker.

IR extended deception at the protocol level by creating predefined sets of responses to packets that could be controlled by a rule set similar to router rules. The IR enables packets to be routed through different interfaces so that the same IP address goes to different networks depending on measurable parameters in the language of the IR. The IR also first introduced mirroring, an effect that is highly successful at causing higher skills attackers to become confused and introduced limited protocol-level deceptions such as dazzlements and "Window zero" responses to force TCP sessions to remain open indefinitely. This particular mechanism had also been implemented at around the same time in special purpose tools. The IR implemented the "Wall" portion of the DWALL technology in a single box, something described in the DWALL patent but first implemented in the IR.

Responder is a Lisp-based tool that handles raw packets directly and uses a combination of a router-like syntax and the ability to add lisp statements at any part of the packet handling process. It also adds hash tables to various fields to increase performance and provides interfaces to higher-level controls so that graphic interfaces and external controls can be applied. The advantage to the Responder technology is that arbitrary changes can be made to packets via the Lisp programming interface, Thus, in addition to emulation of protocol element s associated with various machines and operating systems, Responder can allow arbitrary programmed responses, complex state machines, and interfaces to DTK-like services, all in a single machine that covers arbitrary address spaces. Since it operates at line speed, it can emulate arbitrary network conditions. This includes the ability to model complex infrastructures. The Responder technology also provides playback and packet generation mechanisms to allow the creation of deceptions against local passive sniffers and can coordinate these activities with other deceptions so that it works against proximate attackers as well as distant attackers.

Execution wrappers augment the overall deception mechanisms by creating operating system level deceptions that are invoked whenever a program is executed. The first execution wrapper implementation was done in White Glove Linux and applied to create pairs of computers that acted in concert to provide highly effective deceptions against insiders with systems administrator access. In this particular case, because a bootable CD-based operating system was used, identical configurations could be created on two computers, one with content to be protected, and the other with false content. The execution wrapper was then used to execute unauthorized programs on the second computer. The decision on where to execute a program was based on system state and process lineage, and a series of experimental developments were used to demonstrate that this technology was capable of successfully deceiving systems administrators who tried to exceed their mandate and access content they were not authorized to see. The technology was then applied to a deception in which a Responder was used at the network level to control where attackers were directed based on their behavior and once legitimate users gained access to protected computers, they were again deceived by execution wrappers when they attempted unauthorized usage.

These deceptions were quite successful in the limited experiments undertaken and the combined effects of external and internal deceptions provided a far greater range of options for the deception designer than had previously been available. The advantage of more options is that more error mechanisms can be exploited under better control..

1.4.4 The HoneyNet Project

The HoneyNet project is dedicated to learning about the tools, tactics, and motives of the “blackhat” community and sharing the lessons learned. The primary tool used to gather this information is the Honeynet; a network of production systems designed to be compromised. Unlike most historic honeypots, the Honeynet project is not directed so much at deception to defeat the attacker in the tactical sense as at intelligence gathering for strategic advantage.

This project has been joined by a substantial number of individual researchers and has had substantial success at providing information on widespread attacks, including the detection of large-scale denial of service worms prior to the use of the 'zombies' for attack. At least one Masters thesis is was completed in 2002 based on these results. The Honeynet project has grown over the years into a global effort involving scores of researchers and has included substantial tool development in recent years.

Honeyd is the main line tool of this project. It consists of a program that creates sets of personalities associated with different machiones based on known machine patterns associated with the detection mechanisms of “nmap”, a network mapping program that does active fingerprinting. This is a variation on the D-WALL patent. Like Responder and IR, it can emulate an arbitrary number of hosts by responding to packets and like DTK it can create more in-depth fictions associated with specific services on ports for each of those machines. It also does a passable job of emulating network structures. Honeyd on Open BDS and Arpd in a CD (HOACD) is the implementation of a low-interaction honeypot that runs directly from a CD and stores its logs and configuration files on a hard disk. The “honeydsum.pl” tool turns Honeyd logs into text output and can be used to correlate logs from multiple honeypots. Tools like mydoom.pl and

kuang2.pl provide emulations of systems attacked by specific worms so that attackers who use residual exploits associated with these attacks can be traced.

1.4.5 RIDLR and Software Decoys

The RIDLR is a project launched from Naval Post Graduate School designed to test out the value of deception for detecting and defending against attacks on military information systems. RIDLR has been tested on several occasions at the Naval Post Graduate School. Software decoys were created in another set of projects at Naval Postgraduate school. In this case, an object-oriented architecture was augmented to include fictitious objects designed to provide specific responses to specific attempts to exploit potential system weaknesses. [74]

1.4.6 The Rand Studies

In 1999, RAND completed an initial survey of deceptions in an attempt to understand the issues underlying deceptions for information protection. [18] This effort included a historical study of issues, limited tool development, and limited testing with reasonably skilled attackers. The objective was to scratch the surface of possibilities and assess the value of further explorations. It predominantly explored intelligence related efforts against systems and methods for concealment of content and creation of large volumes of false content. It sought to understand the space of friendly defensive deceptions and gain a handle on what was likely to be effective in the future.

The follow-up RAND study [24] extends the previous results with a set of experiments in the effectiveness of deception against sample forces. They characterize deception as an element of "active network defense". Not surprisingly, they conclude that more elaborate deceptions are more effective, but they also find a high degree of effectiveness for select superficial deceptions against select superficial intelligence probes. They conclude, among other things, that deception can be effective in protection, counterintelligence, against cyber-reconnaissance, and to help to gather data about enemy reconnaissance. This is consistent with previous results that were more speculative. Counter deception issues are also discussed, including (1) structural, (2) strategic, (3) cognitive, (4) deceptive, and (5) overwhelming approaches.

1.4.7 Deception in GOLEM

GOLEM is a system of software “agents” (programs) that are designed to perform goal directed activities with specific behaviors. Because they interact, the researchers who developed these systems experienced the effect of incorrect answers and ultimately came to understand that deceptions could be effective at inducing a wide range of malicious and benevolent behaviors in their system. By exploiting these results they were able to generate helpful responses from otherwise unfriendly programs, showed some mathematical results about their simulation environment, and were able to classify several different sorts of effects. [73]

1.4.8 Older Theoretical Work

One historical and three current theoretical efforts have been undertaken in this area. All are currently quite limited. Cohen looked at a mathematical structure of simple defensive network deceptions in 1999 [7] and concluded that as a counterintelligence tool, network-based deceptions could be of significant value, particularly if the quality of the deceptions could be made good enough. Cohen suggested the use of rerouting methods combined with live systems of the sorts being modeled as yielding the highest fidelity in a deception. He also expressed the limits of fidelity associated with system content, traffic patterns, and user behavior, all of which could be simulated with increasing accuracy for increasing cost. In this paper, networks of up to 64,000 IP addresses were emulated for high quality deceptions using a technology called D-WALL. [66]

Glen Sharlun of the Naval Post Graduate School recently finished a Master's thesis on the effect of deception as a deterrent and as a detection method in large-scale distributed denial of service attacks. Deceptive delays in program response were used by Somayaji to differentiate between

human and automated mechanisms. Error mechanisms were identified for passive and active attack methods and these error mechanisms were used to derive a theoretical approach to systematically creating deceptions that affect the cognitive systems of computers, people, and organizations. [70] This theoretical model describes the methods used to lead attackers through attack graphs with deceptions [71].

1.4.9 Contentions over the use of deception

There is some contention in the world community surrounding the use of these and other deceptive techniques in defense of information systems. The contention seems to be around a few specific issues; (1) the morality of "lying" by presenting a false target for attacks; (2) legal liabilities that might be associated with deceptions; (3) the potential that legitimate users might be deceived and thus waste their time and fall under suspicion, and (4) the need for deceptions as opposed to other "legitimate" approaches to defending those systems.

> Argument 4 is specious on its face. Presumably the market will settle the relative value of different approaches in terms of their utility. In addition, because deceptions systems have proven effective in many arenas, there seems little doubt as to the potential for effective use of deception. Presumable defenders will not have to start telling attackers that they have guessed an invalid user identity before they try a password because, as a deception, this is somehow not legitimate.
>
> Argument 3 is certainly a legitimate concern, but experimentally this has never been a real issue. For large classes of deception systems, the "distance" between legitimate users and the deceptions is so large that they never substantially interact. Significant effort must be undertaken in creating effective deceptions to determine what will have best effect while minimizing potentials for undesired side effects. In this sense, armature approaches to deception are likely to be less effective than those undertaken by experienced professionals, but everyone gets experience somewhere. There is a need for an appropriate place for those who wish to learn to do so in relative safety.
>
> Argument 2 depends on the specifics of the legal climate and the deceptions in use. Clearly there are limits to the use of deception within any present legal framework; however, these limits are relatively easily avoided by prudent application of due diligence with regard to legality within each jurisdiction. A good example was a mirroing with dazzlement approach to defending against worms. Because this crashed the attacking computers, liability was a concern, and after it was shown effective, it was ceased to prevent law suits.
>
> Argument 1, the morality of deception, depends on a social structure that varies greatly and seems to have more to do with presentation and perception than with specific facts. In particular, when presented as a "honeypot", deceptions are widely accepted and often hailed as brilliant, while the same deceptions presented under other names, such as "deceptions", are viewed negatively. To avoid the negative connotation, different verbiage seems to be adequate.

2. Theoretical Results on Deceptions

Deception theory has been undertaken in a number of arenas. While most of the real understanding of deceptions from an implementation point of view surround the notion that deceptions exploit cognitive errors, most of the theoretical work has been oriented in a more mathematical domain. As a result of various research efforts, some interesting issues come to light. There appear to be some features of deception that apply to all of the targets of interest. While the detailed mechanisms underlying these features may differ, commonalities are worthy of note.

2.1 Core Issues

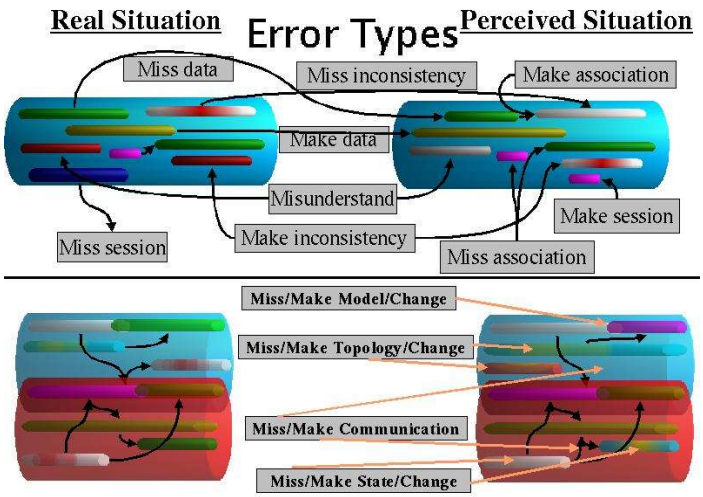
Some core issues seem to recur in most deceptions. They are outlined here as an introduction as it originally appears in [68]. These issues should be addressed in order to assure that deceptions operate effectively and without undue hazard.

Limited Resources lead to Controlled Focus of Attention	By pressuring or taking advantage of pre-existing circumstances focus of attention can be stressed. In addition, focus can be inhibited, enhanced, and through the combination of these, redirected.
All Deception is a Composition of Concealments and Simulations	Concealments inhibit observation while simulations enhance observation. When used in combination they provide the means for redirection.
Memory and Cognitive Structure Force Uncertainty, Predictability, and Novelty	The limits of cognition force the use of rules of thumb as shortcuts to avoid the paralysis of analysis. This provides the means for inducing desired behavior through the discovery and exploitation of these rules of thumb in a manner that restricts or avoids higher level cognition.
Time, timing, and sequence are critical	All deceptions have limits in planning time, time to perform, time till effect, time till discovery, sustainability, and sequences of acts.
Observables Limit Deception	Target, target allies, and deceiver observables limit deception and deception control.
Operational Security is a Requirement	Determining what needs to be kept secret involves a trade off that requires metrics in order to properly address.
Cybernetics and System Resource Limitations	Natural tendencies to retain stability lead to potentially exploitable movement or retention of stability states.
The Recursive Nature of Deception	Recursion between parties leads to uncertainty that cannot be perfectly resolved but that can be approached with an appropriate basis for association to ground truth.
Large Systems are Affected by Small Changes	For organizations and other complex systems, finding the key components to move and finding ways to move them forms a tactic for the selective use of deception to great effect.
Even Simple Deceptions are Often Quite Complex	The complexity of what underlies a deception makes detailed analysis quite a substantial task.
Simple Deceptions are Combined to Form Complex Deceptions	Big deceptions are formed from small sub-deceptions and yet they can be surprisingly effective.
Knowledge of the Target	Knowledge of the target is one of the key elements in effective deception.
Legality	There are legal restrictions on some sorts of deceptions and these must be considered in any implementation.
Modeling Problems	There are many problems associated with forging and using good models of deception.
Unintended Consequences	You may fool your own forces, create mis-associations, and create mis-attributions. Collateral deception has often been observed.
Counterdeception	Target capabilities for counterdeception may result in deceptions being detected.

2.2 Error Models

Passive and active intelligence models have been created for seeking to understand how people and the systems they use to gather information are applied in the information technology arena. These models produced two structures for cognition and cognitive errors. The model in Figure 1 shows error types in a set of models in which the attacker of the system can passively or actively observe a system under attack. In this case, like visual perception is formed from the analysis of sequences of light flashes inducing signals that enter the brain, perception of computer situations is formed by analysis of sequences of observables that flash into other computers, is analyzed by

those computers, and produces depictions for the user. Errors include making and missing data, consistencies, inconsistencies, sessions, and associations. In the active case, where the attacker is able to provide information and see how the defender responds to that information, additional errors include making and missing models, model changes, topologies, topology changes, communications, communications changes, states, and state changes. The target of the deception in the case of a honeypot is an active attacker who can be presented with information that induces errors of these sorts.



For each error type, specific mechanisms have been identified in computers, humans, organizations, and combinations of these, and these mechanisms have been exploited systematically to drive attackers through attack graphs designed by defenders. [71] This goes with the basic theory of deceptions in that the way deceptions can be designed is by (1) identifying error types, (2) identifying and implementing mechanisms that induce those error types, and (3) selectively applying those mechanisms to cause desired effects in the target of the deception. The experiments described later in this paper were used to confirm or refute this underlying theory as well as the specific error mechanisms and the specific deception mechanisms used to induce these sorts of errors. While only a relatively small number of experiments have been performed, the theoretical underpinning appears to be strong and the general methodology has worked effectively when applied systematically.

Figure 1 – Error Types in Network Attacks

2.3 Models of Deception Effectiveness

A mathematical structure was attempted in 1999 for understanding the implications of deception on attacker and defender workload and timing issues. [7] This effort resulted in the characterization of certain classes of deceptions as having the following properties identified in [6]:

- Deception increases the attacker's workload
- Deception allows defenders to better track attacks and respond before attackers succeed
- Deception exhausts attacker resources
- Deception increases the sophistication required for attack
- Deception increases attacker uncertainty

Different deceptions produce different mathematical properties, however, for a class of deceptions involving honeypots and other related decoys, deception can be thought of in terms of their coverage of a space. These notions are based on an implied model of an attacker that was subsequently detailed in [71] using the model provided in figure 2.

In this model, an attacker is assumed to be undertaking an overall attack effort involving intelligence gathering, entries, privilege expansions, and privilege exploitations. The structure of this leads to an attack graph in which deceptions create additional alternatives for the attacker. Specifically, in seeking a target, deception can suppress signals thus causing the attacker to fail to find a real target or, in the case of honeypots and decoys, induce signals to cause the attacker to find false targets. In attempting to differentiate deceptions from non-deceptions, successful honeypots and decoys consume attacker resources, and in some cases cause the erroneous belief that the false targets are real. The result of deceptions that are this successful is that the attacker goes further through the attack tree in the examination of false targets. An additional side effect seen in experiments is that real targets may be misidentified as false targets, thus causing attackers to believe that real systems are in fact honeypots. The model shown in Figure 2 is also recursive and has other properties of interest to the serious student of computer-related deception.

Examples of specific mechanisms that can be applied to driving attackers through these attack graphs are easy to come by. For example, the creation of large numbers of fictitious services and addresses in an Internet Protocol (IP) network creates a large number of cases of finding false targets and, because of the increased cognitive workload, for less detail-oriented attackers, it also causes attackers to miss real targets. This is readily achieved by technologies such as DWALL, the IR, HoneyD, DTK, and Responder. Similarly, mechanisms like execution wrappers have proven effective at causing attackers to transition from a successful position after entry to a deception when they seek to “Exploit Access”. The effect of this technology is that they recursively go down the deceptive attack graph, making the transition from the highest level of “Attack Success” to “Deception Success”.

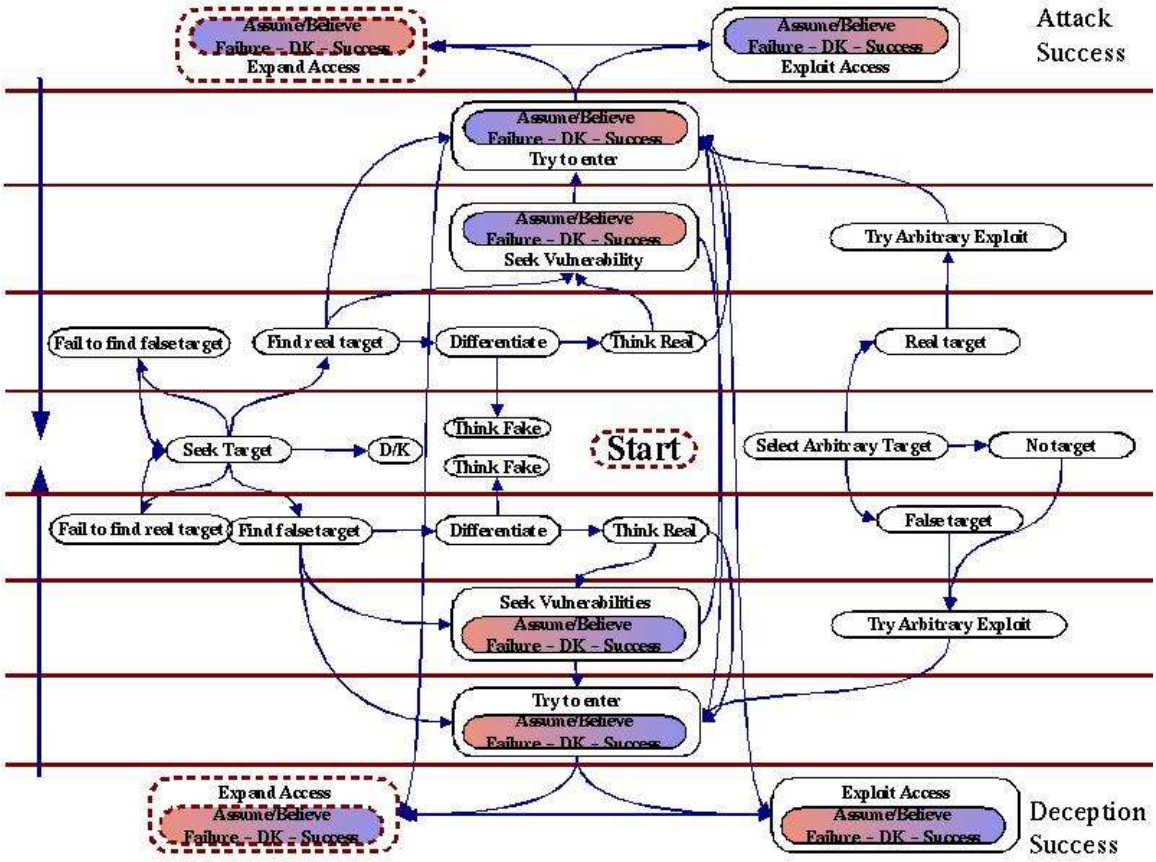


Figure 2 – The Generic Attack Graph with Deception

Progress in the attack graph over time has also proven to be a valuable metric in assessing the effectiveness of defenses of all sorts. While it was first applied to deception experiments where there are positive and negative values associated respectively with increased travel up the real attack graph and increased travel down the deceptive attack graph. Thus in the example above, the attacker went from +4 to level -4 under the execution wrapper, while the network-level deceptions tend to cause attackers to remain at level 0 and -1 for extended periods of time. In non-deception environment, progress can only go in a negative direction under self-deception. The specific error types exploited in the execution wrapper case are missed and made topology, state and state change. The errors made in the network deception cases are missed and made topology, sessions, and associations.

In the mathematical characterizations of deception workload, the effort expended by the attacker depends on the relative number of paths through the attack graph for deceptions and non-deceptions. With no deceptions, all paths are real and the attacker always gains information as they explore the space of real systems. With deceptions in place, a portion of the exploration produces false results. As the total space of attacker options grows large, if far more deceptions than actual systems are presented, the workload of the attacker for detecting real targets and differentiating between real and deception systems increases. Depending on the specifics of the situation, very high workloads can be attained for the attacker. At the same time, the defender who is able to observe attacker activity gains rapid knowledge of the presence of an attacker and their characteristics and can direct further deceptions toward the attacker. Specifically, the characteristics identified with the attacker can be used to present deceptions, even for real services.

Attackers can, in turn, seek to present different characteristics, including characteristics closely associated with legitimate users, in order to make it harder for the deception system to detect them, differentiate between attackers and legitimate users, and increase defender workload. But attackers also have finite resources. As a result, the relative resources of attacker and defender, the number of deceptions vs. non-deceptions, and the time and complexity of attacker and defender efforts play into the overall balance of effort. It turns out that for typical Internet-based intelligence efforts using common tools for network mapping and vulnerability detection, defenders using deceptions have an enormous mathematical advantage. With the addition of rapid detection and response, which the defender gains with deception, the likelihood of attacker success and cost of defense can both be greatly reduced from deceptionless situations.

2.4 Honeypots

While simplistic deceptions used in DTK and the HoneyNet project involve very low fidelity deceptions, typical honeypots involve a small number of high quality deceptions. These systems are typically oriented toward specific target audiences.

- In broad-scale detection, deceptions gain effect by large scale deployment at randomly selected locations in a large space. For example, to rapidly detect widespread computer worms that enter certain classes of systems through random or pseudo-random sweeps of the Internet protocol (IP) address space, a number of systems are deployed at random locations and they await the appearance of malicious activity. If multiple systems detect similar activities, it is very likely to be a widespread attack. The more systems are placed, the sooner the attack will likely be detected, but the timeliness is not linear with the number of systems. Rather, the probability goes up with the number of deceptions placed in proportion to the size of the total space, while the time to detect is a function of the probability of encountering one or more of the deceptions systems as a function of the way the worm spreads. This is the hope of the honeynet project and proposals made to DARPA and other agencies for large-scale deception-based detection arrays for rapid detection of large-scale worms.
- For more targeted deceptions aimed at specific audiences, a different approach is undertaken. For example, the RIDLR project at NPS placed select systems on the Internet with specific characteristics in order to cause those systems to be noticed by

specific audiences. These deceptions are more demanding in terms of deception system fidelity because they typically have to fool human attackers for enough time to gain the advantage desired by the placement of the deception. In one experiment, a system was placed with information on a specific subject known to be of interest to an opposition intelligence agency. The system was populated with specific information and had a regular user population consisting of students who were working on deception-related research. These users had created fictitious identities with specific characteristics of interest and were regularly interacting with each other based on those identities. The deception system includes a specially placed typical but not too obvious vulnerability specifically designed to allow an attacker to enter if they targeted the system. It was identified into Internet search engines by one of its fictitious users and was thus probed by those engines and found in searches by people interested in the specific topics. The execution wrappers systems described above are examples of mechanisms that have been successfully used in high fidelity deceptions oriented toward specific targets.

2.5 Decoys

Decoys are typically thought of as larger-scale, lower fidelity systems intended to change the statistical success rate of tactical attacks. For example, Deception ToolKit, DWALL, the Invisible Router, HoneyD, and Responder are designed to produce large numbers of deceptive services of different characteristics that dominate a search space. The basic idea is to fill the search space of the attacker's intelligence effort with decoys so that detection and differentiation of real targets becomes difficult or expensive. In this approach, the attacker seeking to find a target does a typical sweep of an address space looking for some set of services of interest. DWALL and Responder are also useful for high fidelity deceptions, but these deceptions require far more effort.

Tools like "Nmap" map networks and provide lists of available services, while more sophisticated vulnerability testing tools identify operating system and server types and versions and associate them with specific vulnerabilities. Penetration testing tools go a step further and provide live exploits that allow the user to semi-automatically exploit identified vulnerabilities and do multi-step attack sequences with automated assistance. These tools have specific algorithmic methods of identifying known systems types and vulnerabilities, and the characteristics of the tools are readily identified by targets of their attacks if properly designed for that purpose. The defender can then simulate a variety of operating systems and services using these tools so that the user of the attack tools makes cognitive errors indirectly induced by the exploitation of cognitive errors in their tools. The deceived attacker than proceeds down defender-desired attack graphs while the defender traces the attacks to their source, calls in law enforcement or other response organizations, or feeds false information to the attacker to gain some strategic advantage. In at least one case, defenders included Trojan horse components in software placed in a honeypot with the intent of having that software stolen and used by the attackers. The Trojan horse contained mechanisms that induced covert channels in communication designed to give the so-called defenders an attack capability against the (so-called) attackers' systems.

Of course not all decoys are so high quality. Simple decoys like Deception ToolKit are simple to detect and defeat. Yet after more than seven years of use, they are still effective at detecting and defeating low quality attackers that dominate the attack space. Such tools are completely automatic and inexpensive to operate, don't interfere with normal use, and provide clear detailed indications of the presence of attacks in a timely fashion. While they are ineffective against high skills attackers, they do free up time and effort that would otherwise be spent on less skilled attackers. This is similar to the effectiveness of decoys in military systems. Just as typical chaff defeats many automated heat or radar seeking attack missiles, simple informational deceptions defeat automated attack tools. And just as good pilots are able to see past deceptions like chaff, so skilled information attackers are able to defeat see past deceptions like Deception ToolKit. And just as chaff is still used in defeating missiles despite its limitations, so should simple deceptions be used to defeat automated attack tools despite their limitations. As long as the chaff costs less than the risks it mitigates, it is a good defense, and as long as simple deceptions reduce risk by more than the cost to deploy and operate them, they are good defenses as well.

Higher quality decoys are also worthwhile, but as the quality of the decoy goes up, so does its cost. While some of the more complex decoy systems like DWALL provide more in-depth automation for larger scale deceptions, the cost of these systems is far greater than Deception ToolKit as well. For example, a single DWALL implementation can cost a hundred thousands dollars of initial cost plus substantial operating costs to cover a few tens of thousands of IP addresses. Lower fidelity systems like IR or Responder cost under $10,000 and cover the same sized address space. While Responder and IR can be used to implement the DWALL functions, they also require additional hardware and programming to achieve the same level of fidelity. At some point the benefits of higher fidelity decoys are outweighed by their costs.

2.6 A Model for Deception of Computers

In looking at deceptions against computers it is fundamental to understand that the computer is an automaton. Anthropomorphizing a computer into an intelligent being is a mistake in this context - a self-deception. Fundamentally, deceptions must cause systems to do things differently based on their lack of ability to differentiate a deception from a non-deception. Computers cannot really yet be called “aware” in the sense of people. Therefore, when we use a deception against a computer we are really using a deception against the skills of the human(s) that design, program, and use the computer.

In many ways computers could be better at detecting deceptions than people because of their tremendous logical analysis capability and the fact that the logical processes used by computers are normally quite different than the processes used by people. This provides some level of redundancy and, in general, redundancy is a way to defeat corruption. Fortunately for those of us looking to do defensive deception against automated systems, most of the designers of modern attack technology have a tendency to minimize their programming effort and thus tend not to include a lot of redundancy in their analysis.

People use shortcuts in their programs just as they use shortcuts in their thinking. Their goal is to get to an answer quickly and in many cases without adequate information to make definitive selections. Computer power and memory are limited just like human brain power and memory are limited. In order to make efficient use of resources, people write programs that jump to premature conclusions and fail to completely verify content. In addition, people who observe computer output have a tendency to believe it. Therefore, if we can deceive the automation used by people to make decisions, we may often be able to deceive the users and avoid in-depth analysis.

A good example of this phenomenon is the use of packet sniffers and analyzers by attackers. The analysis tools in widespread use have faults that are not obvious to their users in that they project depictions of sessions even when the supposed sessions are not precisely accurate in the sense of correctly following the protocol specifications. Transmission Control Protocol (TCP) packets, for example, provide ordering and other similar checks, however, deceptions have been successfully used to cause these systems to project incorrect character sequences to their users, providing inaccurate user identification and authentication information for unencrypted terminal sessions. The net effect is that the attacker gets the wrong user identification and password, attempts to log into the system under attack, and is given access to a deception system. The combination of “Make Data” and “Miss Inconsistency” errors by the program and the user cause the deception to be effrective.

Our model for computer deception starts with a model presented in "Structure of Intrusion and Intrusion Detection". [3] In this model, a computer system and its vulnerabilities are described in terms of intrusions at the hardware, device driver, protocol, operating system, library and support function, application, recursive language, and meaning vs. content levels. The levels are all able to interact, but they usually interact hierarchically with each level interacting with the ones just above and below it. This model is depicted in the graphic in Figure 3:

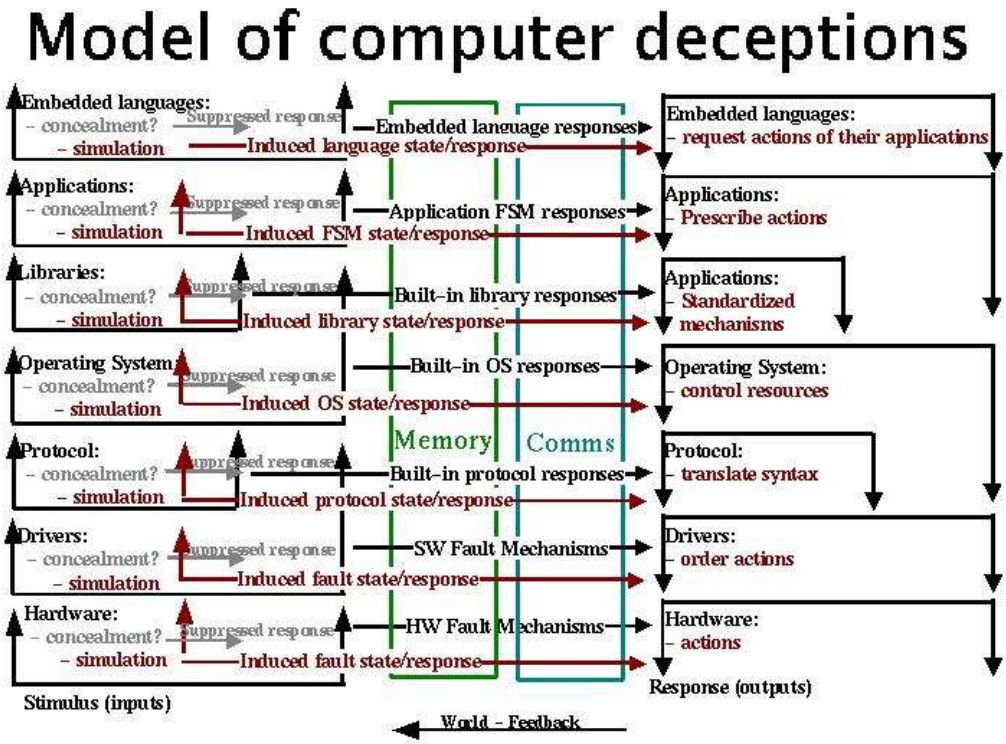


Figure 3 – A Model of Computer Cognitive Failure Mechanisms Leading to Deceptions

This model is based on the notion that at every level of the computer's cognitive hierarchy, signals can either be induced or inhibited. The normal process is shown in black, while inhibitions are shown as grayed out signals, and induced signals are shown in red. All of these affect memory states and processor activities at other, typically adjacent, levels of the cognitive system. Deception detection and response capabilities are key issues in the ability to defend against deceptions so there is a concentration on the limits of detection in the following discussions.

2.6.1 Hardware Level Deceptions

While some honeypots and decoys use hardware level deceptions for local area networks, from remote sites, these deceptions are problematic because the hardware level information associated with systems is not generally available to remote locations.

2.6.2 Driver Level Deceptions

Driver level deceptions are used by some decoys. For example, both the Invisible Router and Responder are able to create protocol disruptions to remote drivers by forcing them to stay engaged in sessions. For large- scale worms and remote network scanners, drivers on attacking systems that strictly follow protocols sometimes are unable to break free of their remote sessions and after attempting more than a small finite number of connections, become permanently stuck and unable to scan further. Typically the programs operating these drivers then fail to make progress and the system or application crashes.

2.6.3 Protocol Level Deceptions

Defensive protocol level deceptions have proven relatively easy to develop and hard to defeat. Deception ToolKit [6] and D-WALL [7] both use protocol level deceptions to great effect and these are relatively simplistic mechanisms compared to what could be devised with substantial time and effort. NoneyD uses a similar mechanism. This appears to be a ripe area for further work. Most intelligence gathering today starts at the protocol level, overrun situations almost universally result in communication with other systems at the protocol level, and insiders generally access other systems in the environment through the protocol level. Most remote driver deceptions are actually protocol level deceptions that occur because protocols are embedded in drivers. They also operate at the protocol level against systems that do not have such driver problems. One of the best examples is the use of mirroring (switching source and destination IP address and port numbers and emitting the input packet on the same interface it arrived on). Mirroring in buffer overrun attacks reflects the original attack against its source. This causes human attackers to attack themselves, sometimes to great effect. If randomization is added toward the end of the packets, automated input buffer overrun attacks tend to crash the remote machines launching the attacks. These defenses have the potential to induce significant liability on the defender who chooses to use them.

2.6.4 Operating System Level Deceptions

To use defensive deception at the target's operating system level requires offensive actions on the part of the deceiver and yields only indirect control over the target's cognitive capability. This has to then be exploited in order to affect deceptions at other levels and this exploitation may be very complex depending on the specific objective of the deception. This is not something done by honeypots or decoys on the market today, however, some honeypots have included software-based Trojan horses designed to attack the attacker by exploiting operating system and application weaknesses. The liability issues are such that this would only be suitable for governments.

2.6.5 Library and Support Function Level Intrusions

Using library functions for defensive deceptions offers great opportunity but, like operating systems, there are limits to the effectiveness of libraries because they are at a level below that used by higher level cognitive functions and thus there is great complexity in producing just the right effects without providing obvious evidence that something is not right. Library weaknesses have been exploited in the same manner as protocol weaknesses to cause attackers to become temporarily disabled when their intelligence software becomes unable to handle the responses.

2.6.6 Application Level Deceptions

Applications provide many new opportunities for deceptions. The apparent user interface languages offer syntax and semantics that may be exploited while the actual user interface languages may differ from the apparent languages because of programming errors, back doors, and unanticipated interactions. Internal semantics may be in error, may fail to take all possible situations into account, or there may be interactions with other programs in the environment or with state information held by the operating environment. They always trust the data they receive so that false content is easily generated and efficient. These include most intelligence tools, exploits, and other tools and techniques used by severe threats. Known attack detection tools and anomaly detection have been applied at the application level with limited success. Network detection mechanisms also tend to operate at the application level for select known application vulnerabilities. A good example is the presentation of false information in response to application-generated network probes. The responses generate false information which reaches the user appearing to be accurate and in keeping with the normal operation of the tool. This is the class of deceptions exploited in most of the experiments in leading attackers through attack graphs.

Application level defensive deceptions are very likely to be a major area of interest because applications tend to be driven more by time to market than by surety and because applications tend to directly influence the decision processes made by attackers. For example, a defensive deception would typically cause a network scanner to make wrong decisions and report wrong

results to the intelligence operative using it. Similarly, an application level deception might be used to cause a system that is overrun to act on the wrong data. For systems administrators the problem is somewhat more complex and it is less likely that application-level deceptions will work against them.

2.6.7 Recursive Languages in the Operating Environment

Recursive languages are used in many applications including many intelligence and systems administration applications. In cases where this can be defined or understood or cases where the recursive language itself acts as the application, deceptions against these recursive languages should work in much the same manner as deceptions against the applications themselves. This is suitable only to government-level operations because of the potential liabilities associated with its use.

2.7 Commentary

Unlike people, computers don't typically have egos, but they do have built-in expectations and in some cases automatically seek to attain 'goals'. If those expectations and goals can be met or encouraged while carrying out the deception, the computers will fall prey just as people do.

In order to be very successful at defeating computers through deception, there are three basic approaches. One approach is to create as high a fidelity deception as you can and hope that the computer will be fooled. Another is to understand what data the computer is collecting and how it analyzes the data provided to it. The third is to alter the function of the computer to comply with your needs. The high fidelity approach can be quite expensive but should not be abandoned out of hand. At the same time, the approach of understanding enemy tools can never be done definitively without a tremendous intelligence capability. The modification of cognition approach requires an offensive capability that is not always available and is quite often illegal, but all three avenues appear to be worth pursuing.

High Fidelity: High fidelity deception of computers with regard to their assessment, analysis, and use against other computers tends to be fairly easy to accomplish today using tools like the deception wall (D-WALL) [7], the invisible router (IR), and Responder in conjunction with tools like execution wrappers. While this is effective in the generic sense, for specific systems, additional effort must be made to create the internal system conditions indicative of the desired deception environment. This can be quite costly. These deceptions tend to operate at a protocol level and are augmented by other technologies to affect other levels of deception.

Defeating Specific Tools: Many specific tools are defeated by specific deception techniques. For example, nmap and similar scans of a network seeking out services to exploit are easily defeated by tools like the Deception ToolKit [6] and HoneyD. More specific attack tools such as Back Orafice (BO) can be directly countered by specific emulators such as "NoBO" - a PC-based tool that emulates a system that has already been subverted with BO. Some deception systems work against substantial classes of attack tools. HoneyD and the HoneyNet project attempts to create specific deceptions for widely spread worms.

Modifying Function: Modifying the function of computers is relatively easy to do and is commonly used in attacks. The question of legality aside, the technical aspects of modifying function for defense falls into the area of counterattack and is thus not a purely defensive operation. The basic plan is to gain access, expand privileges, induce desired changes for ultimate compliance, leave those changes in place, periodically verify proper operation, and exploit as desired. In some cases privileges gained in one system are used to attack other systems as well. Modified function is particularly useful for getting feedback on target cognition.

The intelligence requirements of defeating specific tools may be substantial, but the extremely low cost of such defenses makes them appealing. Against off-the-Internet attack tools, these defenses are commonly effective and, at a minimum, increase the cost of attack far more than they affect the cost of defense. Unfortunately, for more severe threats, such as insiders, overrun situations, and intelligence organizations, these defenses are often inadequate. They are almost certain to be detected and avoided by an attacker with skills and access of this sort. Nevertheless,

from a standpoint of defeating the automation used by these types of attackers, relatively low-level deceptions have proven effective. In the case of modifying target systems, the problems become more severe in the case of more severe threats. Insiders are using your systems, so modifying them to allow for deception allows for self-deception and enemy deception of you. For overrun conditions you rarely have access to the target system, so unless you can do very rapid and automated modification, this tactic will likely fail. For intelligence operations this requires that you defeat an intelligence organization one of whose tasks is to deceive you. The implications are unpleasant and inadequate study has been made in this area to make definitive decisions.

There is a general method of deception against computer systems being used to launch fully automated attacks against other computer systems. The general method is to analyze the attacking system (the target) in terms of its use of responses from the defender and create sequences of responses that emulate the desired responses to the target. Because all such mechanisms published or widely used today are quite finite and relatively simplistic, with substantial knowledge of the attack mechanism, it is relatively easy to create a low-quality deception that will be effective. It is noteworthy, for example, that the Deception ToolKit [6], which was made publicly available in source form in 1998, is still almost completely effective against automated intelligence tools attempting to detect vulnerabilities. It seems that the widely used attack tools are not yet being designed to detect and counter deception.

That is not to say that red teams and intelligence agencies are not beginning to start to look at this issue. For example, in private conversations with defenders against select elite red teams the question often comes up of how to defeat the attackers when they undergo a substantial intelligence effort directed at defeating their attempts at deceptive defense. The answer is to increase the fidelity of the deception. This has associated costs, but as the attack tools designed to counter deception improve, so will the requirement for higher fidelity in deceptions.

2.8 Effects of Deceptions on Human Attackers

Attackers facing deception defenses do not go unscathed. In early experiments with deception defenses several results indicated that attackers were negatively impacted. Impacts included reduction in group cohesion, reduced desire to participate in attack activities, reduce enjoyment of activities, increased backtracking even when not under deception, and reduction in performance levels. [71] There was even evidence that one high quality attack team became unable to perform attacks after having been exposed to deception defenses. Even a year later they had problems carrying out effective attacks because they were constantly concerned that they might be under deception. In the section of this article on experiments, more details will be provided on these results. What appears to be clear at this time is that the cognitive mechanisms used for tactical deception are not the only mechanisms at play. Long term effects of deception on a strategic level are not yet as well understood.

2.9 Models of Deception of More Complex Systems

Larger cognitive systems can me modeled as being built up from smaller cognitive subsystems through some composition mechanism. Using these combined models we may analyze and create larger scale deceptions. To date there is no really good theory of composition for these sorts of systems and attempts to build theories of composition for security properties of even relatively simple computer networks have proven rather difficult. We can also take a top-down approach, but without the ability to link top-level objectives to bottom-level capabilities and without metrics for comparing alternatives, the problem space grows rapidly and results cannot be meaningfully compared. Unfortunately, honeypots and decoys are not oriented toward group deceptions, so the work in this area does not apply to these systems.

2.9.1 Criminal Honeypots and Decoys

Criminals have moved to the Internet environment in large numbers and use deception as a fundamental part of their efforts to commit crimes and conceal their identities from law enforcement. While the specific examples are too numerous to list, there are some common

threads, among them that the same criminal activities that have historically worked person to person are being carried out over the Internet with great success.

Identity theft is one of the more common deceptions based on attacking computers. In this case, computers are mined for data regarding an individual and that individual's identity is taken over by the criminal who then commits crimes under the assumed name. The innocent victim of the identity theft is often blamed for the crimes until they prove themselves innocent. Honeypots are commonly used in these and similar deceptions.

Typically a criminal will create a honeypot to collect data on individuals and use a range of deceptive techniques to steer potential victims to the deception. Child exploitation is commonly carried out by creating friends under the fiction of being the same age and sex as the victim. Typically a 40 year old pedophile will engage a child and entice them into a meeting outside the home. In some cases there have been resulting kidnappings, rapes, and even murders. Some of these individuals create child or exploit friendly sites to lure children in.

Larger scale deceptions have also been carried out over the Internet. For example, one of the common methods is to engage a set of 'shills' who make different points toward the same goal in a given forum. These shills are a form of decoys. While the forum is generally promoted as being even handed and fair, the reality is that anyone who says something negative about a particular product or competitor will get lambasted. This has the social effect of causing distrust of the dissenter and furthering the goals of the product maker. The deception is that the seemingly independent members are really part of the same team, or in some cases, the same person. In another example, a student at a California university invested in derivatives of a stock and then made false postings to a financial forum that drove down the price. The net effect was a multi-million dollar profit for the student and the near collapse of the stock. This is another example of a decoy.

The largest scale computer deceptions tend to be the result of computer viruses. Like the mass hysteria of a financial bubble, computer viruses can cause entire networks of computers to act as a rampaging group. It turns out that the most successful viruses today use human behavioral characteristics to induce the operator to foolishly run the virus which, on its own, could not reproduce. They typically send an email with an infected program as an attachment. If the infected program is run it then sends itself in email to other users this user communicates with, and so forth. The deception is the method that convinces the user to run the infected program. To do this, the program might be given an enticing name, or the message may seem like it was really from a friend asking the user to look at something, or perhaps the program is simply masked so as to simulate a normal document.

3. Experiments and the Need for an Experimental Basis

One of the more difficult things to accomplish in the deception arena is meaningful experiments. While a few authors have published experimental results in information protection, far fewer have attempted to use meaningful social science methodologies in these experiments or to provide enough testing to understand real situations. This may be because of the difficulty and high cost of each such experiment and the lack of funding and motivation for such efforts. This is a critical need for future work.

If one thing is clear it is the fact that too few experiments have been done to understand how deception works in defense of computer systems and, more generally, too few controlled experiments have been done to understand the computer attack and defense processes and to characterize them. Without a better empirical basis, it will be hard to make scientific conclusions about such efforts. While anecdotal data can be used to produce many interesting statistics, the scientific utility of those statistics is very limited because they tend to reflect only those examples that people thought worthy of calling out.

Repeatability is also an issue in experiments. While the experiments carried out at Sandia were readily repeated, initial conditions in social experiments are non-trivial to attain. But even more importantly, nobody has apparently sought to do repetitions of experiments under similar conditions or with similar metrics. For example, some experiment to determine the effectiveness

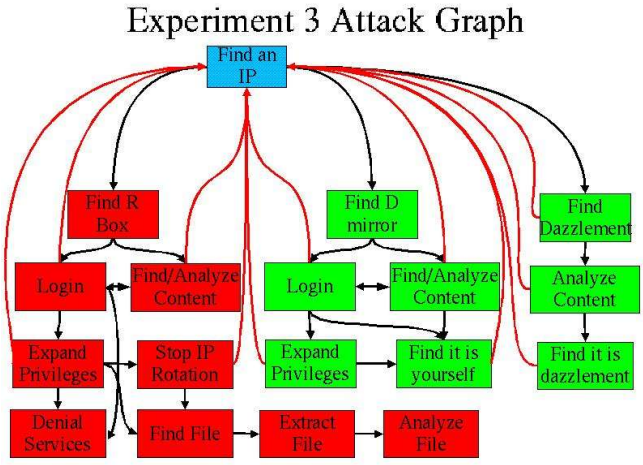
of address rotation were carried out but, despite the fact that address rotation experiments were carried out in the studies described here, the same methodologies were not use in the subsequent experiments, so no direct comparison could be undertaken. In many cases, the expectations of sponsors are that defenses will be perfect or they are not worth using. But deception defenses are essentially never perfect nor can they ever be. They change the characteristics of the search space, but they do not make successful attack impossible. Another major problem is that many experiments tend to measure ill defined things, presumably with the intent of proving a technique to be effective. But experiments that are scientific in nature must seek to refute or confirm specific hypotheses, and they must be measured using some metric that can be fairly measured and independently reviewed.

3.1 Experiments to Date

From the time of the first published results on honeypots, the total number of published experiments performed in this area appears to be very limited. While there have been hundreds of published experiments by scores of authors in the area of human deception, refereed articles on computer deception experiments can be counted on one hand.

3.1.1 Experiments on Test Subjects at Sandia National Laboratories

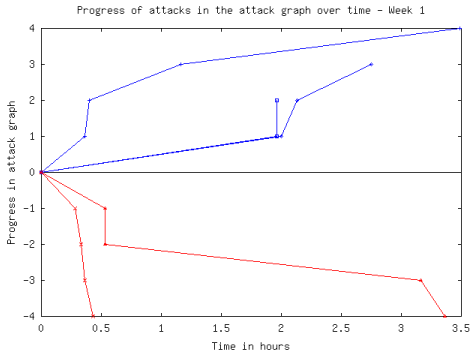
Originally, a few examples of real world effects of deception were provided, **[6]** but no scientific studies of the effects of deception on test subjects were performed. While there was a mathematical analysis of the statistics of deception in a networked environment, there was no empirical data to confirm or refute these results. **[7]** Subsequent experiments [71][72] produced a series of results that have not been independently verified but appear to be accurate based on the available data. In these experiments, forensically sound images of systems and configurations were used to create repeatable configurations that were presented to groups of attackers.



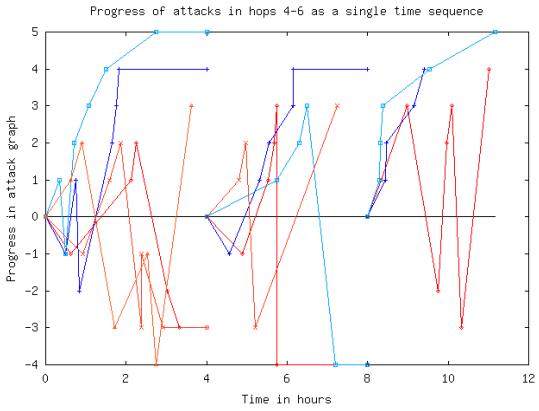
These attack groups were given specific goals for their efforts and were measured by a number of metrics using a combination of observations by experiment monitors, videotaping of sessions which were analyzed, and forms that were filled out as individuals and then as a group at the end of each 4-hour session.

Attack progress was measured over time relative to an attack graph with progress toward the deception (in green) indicated as negative progress and progress toward the real objective (in red) indicated as positive progress. These were all open ended experiments designed so that the attack group would never be able to complete the task but so that progress could be measured. An example result shows attackers not under deception in blue and attackers under deception in red.

In the example provided here, the deception was extremely effective, but it was not as effective in all examples. Nevertheless, deception was shown to be very effective in all of the experiments with attackers generally taking longer to make progress and making less progress over time under deception than attackers not under deception. But results were far more interesting than this when repetition of a single experiment was undertaken with the same groups for week after week.



In this sequence of experiments, the same attack groups were run through the same situation for three weeks in a row. After the first week, one of the groups undergoing deception decided they did not wish to continue their work in this area. The experimenters allowed them to resign, thinking that this was to be expected after 4 straight weeks of four hour sessions. But the next week, the same thing happened again with a second attack group undergoing deception. By the end of the third week of this sequence, out of 25 original attackers, 10 in two groups under no deception and 15 in three groups under deception, only 11 remained. Of the 11 remaining attackers, only 2 were under deception. Subsequent analysis of forms filled out by the groups and behavioral analysis of the motions and actions of individuals indicated that a number of psychological factors were present. In particular, groups under deception started out enjoying the work equally with other groups but ended up enjoying the work less, maintained less group cohesion, reduced trust in leadership, and generally ended up going slower and slower over time, despite the learning that was successful at improving progress over time for groups not under deception. This seems to be consistent with results on learning in children where inconsistent feedback reduces performance.



3.1.2 The HoneyNet Project

The HoneyNet Project [43] is a substantial effort aimed at placing deception systems in the open environment for detection and tracking of attack techniques. As such, they have been largely effective at luring attackers. These lures are real systems placed on the Internet with the purpose of being attacked so that attack methods can be tracked and assessed. As deceptions, the only thing deceptive about them is that they are being watched more closely than would otherwise be apparent and known faults are intentionally not being fixed to allow attacks to proceed. These are highly effective at allowing attackers to enter because they are extremely high fidelity, but only for the purpose they are intended to provide. They do not, for example, include any user behaviors or content of interest. They are quite effective at creating sites that can be exploited for attack of other sites. For all of the potential benefit, however, the HoneyNet project has not performed any controlled experiments to understand the issues of deception effectiveness. In addition, over time the attackers appear to have learned about honeypots and now many of them steer clear of these systems by using indicators of honeypot computers as differentiators for their attacks. For example, they look for user presence in the computers and processes reminiscent of normal user behavior. These deceptions have not apparently been adapted quickly enough to ward off these attackers by simulating a user population.

3.1.3 Red Teaming Experiments

Red teaming (i.e., finding vulnerabilities at the request of defenders) [64] has been performed by many groups for quite some time. The advantage of red teaming is that it provides a relatively realistic example of an attempted attack. The disadvantage is that it tends to be somewhat

artificial and reflective of only a single run at the problem. Real systems get attacked over time by a wide range of attackers with different skill sets and approaches. While many red teaming exercises have been performed, these tend not to provide the scientific data desired in the area of defensive deceptions because they have not historically been oriented toward this sort of defense.

Several red teaming experiments against simplistic defenses were performed under a DARPA research grant in 2000 and these showed that sophisticated red teams were able to rapidly detect and defeat simplistic deceptions. These experiments were performed in a proximity- only case and used static deceptions of the same sort as provided by Deception ToolKit. As a result this was a best case scenario for the attackers. Unfortunately the experimental technique and data from these experiments was poor and inadequate funding and attention was paid to detail. Defenders apparently failed to even provide false traffic for these conditions, a necessity in creating effective deceptions against proximate attackers, and a technique that was used in the Sandia experiments when proximate or enveloped attackers were in use. Only distant attacker models can possibly be effective under these conditions. Nevertheless, these results should be viewed as a cautionary note to the use of low quality deceptions against high quality attackers and should lead to further research into the range of effectiveness of different methods for different situations.

3.1.4 Rand Experiments

War games played out by armed services tend to ignore issues of information system attacks because the exercises are quite expensive and by successfully attacking information systems that comprise command and control capabilities, many of the other purposes of these war games are defeated. While many recognize that the need to realistically portray effects is important, we could say the same thing about nuclear weapons, but that doesn't justify dropping them on our forces for the practice value.

The most definitive experiments to date that we were able to find on the effectiveness of low-quality computer deceptions against high quality computer assisted human attackers were performed by RAND. [24] Their experiments with fairly generic deceptions operated against high quality intelligence agency attackers demonstrated substantial effectiveness for short periods of time. This implies that under certain conditions (i.e., short time frames, high tension, no predisposition to consider deceptions, etc.) these deceptions may be effective.

3.2 Experiments We Believe Are Needed At This Time

The total number of controlled experimental runs to date involving deception in computer networks appear to be less than 50, and the number involving the use of deceptions for defense are limited to the 10 or so from the RAND study and 35 from the Sandia studies. Furthermore, the RAND studies did not use control groups or other methods to differentiate the effectiveness of deceptions. Clearly there is not enough experimental data enough to gain much in the way of knowledge and, just as clearly, many more experiments are required in order to gain a sound understanding of the issues underlying deception for defense.

The clear solution to this dilemma is the creation of a set of experiments in which we use social science methodologies to create, run, and evaluate a substantial set of parameters that provide us with better understanding and specific metrics and accuracy results in this area. In order for this to be effective, we must not only create defenses, but also come to understand how attackers work and think. For this reason, we will need to create red teaming experiments in which we study both the attackers and the effects of defenses on the attackers. In addition, in order to isolate the effects of deception, we need to create control groups, and experiments with double blinded data collection. While the Sandia studies did this and their results are interesting, they are not adequate to draw strong or statistically valid conclusions, particularly in light of the results from subsequent DARPA studies without these controls.

4. Summary, Conclusions, and Further Work

This article has summarized a great deal of information on the history of honeypots and decoys for use in defense of computer systems. While there is a great deal to know about how deception has been used in the past, it seems quite clear that there will be far more to know about deception in the future. The information protection field has an increasingly pressing need for innovations that change the balance between attack and defense. It is clear from what we already know that deception techniques have the demonstrated ability to increase attacker workload and reduce attacker effectiveness, while decreasing defender effort required for detection and providing substantial increases in defender understanding of attacker capabilities and intent.

Modern defensive computer deceptions are in their infancy, but they are moderately effective, even in this simplistic state. The necessary breakthrough that will turn these basic deception techniques and technologies into viable long-term defenses is the linkage of social sciences research with technical development. Specifically, we need to measure the effects and known characteristics of deceptions on the systems comprised of people and their information technology to create, understand, and exploit the psychological and physiological bases for the effectiveness of deceptions. The empirical basis for effective deception in other arenas is simply not available in the information protection arena today, and in order to attain it, there is a crying need for extensive experimentation in this arena.

To a large extent this work has been facilitated by the extensive literature on human and animal deception that has been generated over a long period of time. In recent years, the experimental evidence has accumulated to the point where there is a certain degree of general agreement in the part of the scientific community that studies deception, about many of the underlying mechanisms, the character of deception, the issues in deception detection, and the facets that require further research. These same results and experimental techniques need to be applied to deception for information protection if we are to become designers of effective and reliable deceptions.

The most critical work that must be done in order to make progress is the systematic study of the effectiveness of deception techniques against combined systems with people and computers. This goes hand in hand with experiments on how to counter deceptions and the theoretical and practical limits of deceptions and deception technologies. In addition, codification of prior rules of engagement, the creation of simulation systems and expert systems for analysis of deceptions sequences, and a wide range of related work would clearly be beneficial as a means to apply the results of experiments once empirical results are available.

References

[1] The Boyd Cycle, also known as the observe, orient, decide, act (OODA) loop is described in many articles including; "Boyd Cycle Theory in the Context of Non-Cooperative Games: Implications for Libraries", "The Strategy of the Fighter Pilot", and "Decision Making".
[2] David Lambert, "A Cognitive Model for Exposition of Human Deception and Counter-deception" (NOSC Technical Report 1076 - October, 1987).
[3] Fred Cohen, "The Structure of Intrusion and Intrusion Detection", May 16, 2000, http://all.net/ (InfoSec Baseline Studies)
[4] Fred Cohen, "A Theory of Strategic Games with Uncommon Objectives"
[5] Fred Cohen, "Simulating Cyber Attacks, Defenses, and Consequences", IFIP TC-11, Computers and Security, 1999.
[6] F. Cohen, "A Note on the Role of Deception in Information Protection", Computers and Security 1999.
[7] F. Cohen, "A Mathematical Structure of Simple Defensive Network Deceptions", 1999, http://all.net (InfoSec Baseline Studies).
[8] James F. Dunnigan and Albert A. Nofi, "Victory and Deceipt: Dirty Tricks at War", William Morrow and Co., New York, NY, 1995.
[9] F. Cohen, "Managing Network Security: What does it do behind your back?", July, 2000, Network Security Management Magazine.
[10] Field Manual 90-02: Battlefield Deception, 1998.
[11] Bart Whaley, "Stratagem: Deception and Surprise in War", Cambridge: MIT Center for International Studies. 1969
[12] Chuck Whitlock, "Scam School", MacMillan, 1997.

[13] Bob Fellows, "Easily Fooled", Mind Matters, PO Box 16557, Minneapolis, MN 55416, 2000
[14] Thomas Gilovich, "How We Know What Isn't So: The fallibility of human reason in everyday life", Free Press, NY, 1991
[15] Al Seckel, "The Art of Optical Illusions", Carlton Books, 2000.
[16] Colonel Michael Dewar, "The Art of Deception in Warfare", David and Charles Military Books, 1989.
[17] William L. Griego, "Deception - A 'Systematic Analytic' Approach", (slides from 1978, 1983)
[18] Scott Gerwehr, Jeff Rothenberg, and Robert H. Anderson, "An Arsenal of Deceptions for INFOSEC (OUO)", PM-1167- NSA, October, 1999, RAND National Defense Research Institute Project Memorandum.
[19] Fred Cohen, "Deception Toolkit", March, 1998, available at http://all.net/
[20] Bill Cheswick, Steve Bellovin, Diana D'Angelo, and Paul Glick, "An Evening with Berferd" - followed by S. M. Bellovin. "There Be Dragons". Proceedings of the Third Usenix UNIX Security Symposium. Baltimore (September 1992).
[21] F. Cohen, "Internet Holes - Internet Lightning Rods", Network Security Magazine, July, 1996.
[22] F. Cohen, Operating System Protection Through Program Evolution Computers and Security 1992.
[23] F. Cohen, A Note On Distributed Coordinated Attacks, Computers and Security, 1996.
[24] Scott Gerwehr, Robert Weissler, Jamison Jo Medby, Robert H. Anderson, Jeff Rothenberg, "Employing Deception in Information Systems to Thwart Adversary Reconnaissance- Phase Activities (OUO)", PM-1124- NSA, Novermber 2000, RAND National Defense Research Institute.
[25] Robert E. Huber, "Information Warfare: Opportunity Born of Necessity", News Briefs, September- October 1983, Vol. IX, Num. 5, "Systems Technology" (Sperry Univac) pp 14- 21.
[26] Knowledge Systems Corporation, "C3CM Planning Analyzer: Functional Description (Draft) First Update", RADC/COAD Contract F30602- 87- C- 0103, December 12, 1987.
[27] John J. Ratey, M.D., "A User's Guide to the Brain", Pantheon Books, 2001. [In contrast, the auditory nerve only has about 25,000 nerve fibers. Information must be assessed beginning in the ear itself, guided by the brain. "Evidence that our brains continually shape what we hear lies in the fact that there are more neuronal networks extending from the brain to the ears than there are coming from the ears to the brain." [27] (p. 93)]
[28] Sun Tzu, "The Art of War", (Translated by James Clavell), Dell Publishing, New York, NY 10036 (1983).
[29] Gordon Stein, "Encyclopedia of Hoaxes", Gale Research, Inc, 1993, p. 293.
[30] Fay Faron, "Rip- Off: a writer's guide to crimes of deception", Writers Digest Books, 1998, Cinn, OH.
[31] Richard J. Robertson and William T. Powers, Editors, "Introduction to Modern Psychology, The Control- Theory View". The Control Systems Group, Inc., Gravel Switch, Kentucky, 1990.
[32] Charles K. West, "The Social and Psychological Distortion of Information", Nelson- Hall, Chicago, 1981.
[33] Chester R. Karrass, "The Negotiating Game", Thomas A. Crowell, New York, 1970.
[34] Robert B. Cialdini, "Influence: Science and Practice", Allyn and Bacon, Boston, 2001.
[35] Robert W. Mitchell and Nicholas S. Thompson, "DECEPTION: Perspectives on human and nonhuman deceipt", SUNY Press, 1986, NY.
[36] Donald D. Hoffman, "Visual Intelligence: How We Create What We See", Norton, 1998, NY.
[37] Charles Handy, "Understanding Organizations", Oxford University Press, NY, 1993. img35.jpg
[38] National Research Council, "Modeling Human and Organizational Behavior", National Academy Press, Washington, DC, 1998.
[39] Bill Cheswick, An Evening with Berferd, 1991.
[40] Fred Cohen, "The Unpredictability Defense", Managing Network Security, April, 1998.
[41] David Kahn, "The Code Breakers", Macmillan Press, New York, 1967
[42] Norbert Weiner, "Cybernetics", 1954?
[43] The HoneyNet Project web site (www.honeynet.org).
[44] Tom Keaton, "A History of Warfare", Vintage Books, NY, 1993
[45] Andrew Wilson, "The Bomb and The Computer", Delacorte Press, NY, 1968.
[46] Robert Greene, "The 48 Laws of Power", Penguin Books, New York 1998
[47] Diana Deutsch, "Musical Illusions and Paradoxes", Philomel, La Jolla, CA 1995.
[48] Fred Cohen Cynthia Phillips, Laura Painton Swiler, Timothy Gaylor, Patricia Leary, Fran Rupley, Richard Isler, and Eli Dart, "A Preliminary Classification Scheme for Information System Threats, Attacks, and Defenses; A Cause and Effect Model; and Some Analysis Based on That Model", The Encyclopedia of Computer Science and Technology, 1999.
[49] Richards J. Heuer, Jr., "Psychology of Intelligence Analysis", History Staff Center for the Study of Intelligence Central Intelligence Agency 1999.

[50] Aldert Vrij, "Detecting Lies and Deceipt", Wiley, New York, NY, 2000.
[51] National Technical Baseline, "Intrusion Detection and Response", Lawrence Livermore National Laboratory, Sandia National Laboratories, December, 1996
[52] Various documents, A list of documents related to MKULTRA can be found over the Internet.
[53] Kalbfleisch, Pamela J. The language of detecting deceit. Journal of Language & Social Psychology, Dec94, Vol. 13 Issue 4, p469, 28p, 1 chart [Provides information on the study of language strategies that are used to detect deceptive communication in interpersonal interactions. Classification of the typology; Strategies and implementation tactics; Discussions on deception detection techniques; Conclusion.]
[54] Colonel John Hughes- Wilson, "Military Intelligence Blunders", Carol & Graf, NY, 1999
[55] John Keegan, "A History of Warfare", Vintage Books, NY 1993.
[56] Charles Mackay, "Extraordinary Popular Delusions and the Madness of Crowds", Templeton Publications, 1989 (originally Richard Bently Publishers, London, 1841)
[57] Donald Danial and Katherine Herbig, ed. "Strategic Military Deception", Pergamon Books, 1982.
[58] Western Systems Coordinating Council WSCC Preliminary System Disturbance Report Aug 10, 1996 - DRAFT [This report details the August 10, 1996 major system disturbance that separated the Western Systems Coordinating Council system into 4 islands, interrupting service to 7.5 million customers for periods ranging from several minutes to nearly six hours.]
[59] Bob Pekarske. Restoration in a Flash- - - Using DS3 Cross- connects, Telephony. September 10, 1990. [This paper describes the techniques used to compensate for network failures in certain telephone switching systems in a matter of a millisecond. The paper points out that without this rapid response, the failed node would cause other nodes to fail, causing a domino effect on the entire national communications networks.]
[60] Mimi Ito, "Cybernetic Fantasies: Extended Selfhood in a Virtual Community", 1993.
[61] Mark Peace, "Dissertation: A Chatroom Ethnography", May 2000
[62] Daniel Chandler, "Personal Home Pages and the Construction of Identities on the Web", 2001
[63] Fred Cohen, "Understanding Viruses Bio- logically", Network Security Magazine, Aug, 2000.
[64] Fred Cohen, "Red Teaming and Other Agressive Auditing Techniques", Managing Network Security", March, 1998.
[65] SSCSD Tactical DecisionMaking Under Stress, SPAWAR Systems Center.
[66] Fred Cohen, "Method and Aparatus for Network Deception/Emulation", International Patent Application No PCT/US00/31295, Filed Octoboer 26, 2000.
[67] Heidi Vanderheiden, Boston University "Gender swapping on the Net?", http://web.aq.org/~tigris/loci- virtualtherapy.html
[68] Fred Cohen, Dave Lambert, Charles Preston, Nina Berry, Corbin Stewart, and Eric Thomas, "A Framework for Deception ", available at http://all.net/ under "Deception for Protection".
[69] Fred Cohen, "Responder Manual", available at http://all.net/ under "White Glove Distributions".
[70] Fred Cohen and Deanna Koike, "Errors in the Perception of Computer- Related Information", Jan 12, 2003 http://all.net/journal/deception/Errors/Errors.html and pending publication in IFIP TC- 11 "Computers and Security".
[71] Fred Cohen and Deana Koike, "Leading Attackers Through Attack Graphs with Deceptions ", IEEE Information Assurance Workshop, June 10, 2004, West Point, NY. Also available at: http://all.net/journal/deception/Agraph/Agraph.html
[72] Fred Cohen, Irwin Marin, Jeanne Sappington, Corbin Stewart, and Eric Thomas, "Red Teaming Experiments with Deception Technologies ", available at http://all.net/journal/deception/experiments/experiments.html
[73] Cristano Castelfranhi, Rino Falcone, and Fiorella de Rosis, "Deceiving in GOLEM: how to strategically pilfer help ", 1998, available at http://www.istc.cnr.it/T3/download/aamas1998/Castelfranchi- et- alii.pdf
[74] James B. Michael, Neil C. Rowe, Hy S. Rothstein, Mikhail Auguston, Doron Drusinsky, and Richard D. Riehle, "Phase I Report on Intelligent Software Decoys: Technical Feasibility and Institutional Issues in the Context of Homeland Security", Naval Postgraduate School, Monterey, CA. 10 December, 2002.