Chapter XIII
Deception in Defense of Computer Systems from Cyber Attack

Neil C. Rowe
U.S. Naval Postgraduate School, USA

ABSTRACT

While computer systems can be quite susceptible to deception by attackers, deception by defenders has increasingly been investigated in recent years. Military history has classic examples of defensive deceptions, but not all tactics and strategies have analogies in cyberspace. Honeypots are the most important example today; they are decoy computer systems designed to encourage attacks to collect data about attack methods. We examine the opportunities for deception in honeypots, and then opportunities for deception in ordinary computer systems by tactics like fake information, false delays, false error messages, and identity deception. We conclude with possible strategic deceptions.

INTRODUCTION

Defense from cyber attacks (exploits) in cyberspace is difficult because this kind of warfare is inherently asymmetric with the advantage to the attacker. The attacker can choose the time, place, and methods with little warning to the defender. Thus a multilayered defense (defense in depth) is important (Tirenin & Faatz, 1999). Securing one's cyberspace assets by access controls and authentication methods is the first line of defense, but other strategies and tactics from conventional warfare are also valuable, including deception.

Dunnigan and Nofi (2001) provide a useful taxonomy of nine kinds of military deception similar to several other published ones: concealment, camou-

flage, disinformation, ruses, displays, demonstrations, feints, lies, and manipulation of the adversary by insight into their reasoning and goals. Rowe and Rothstein (2004) propose an alternative taxonomy based on case theory from linguistics. Table 1 shows those categories of deceptions they argue are feasible for defense from cyber attack, with revised assessments of suitability on a scale of 1 (unsuitable) to 10 (suitable). Some of these deceptions also can be used in a "second-order" way, after initial deceptions have been detected by the adversary. An example is creating inept deceptions with obviously false error messages, while also modifying attacker files in a subtle way.

HONEYPOTS

Honeypots are the best-known example of defensive deception in cyberspace (The Honeynet Project, 2004; Spitzner, 2003). These computer systems serve no purpose besides collecting data about attacks on them. That means they have no legitimate users other than system administrator; anyone else who uses them is inherently suspicious. Honeypots record all their activity in secure audit files for later analysis, and the lack of legitimate traffic means this is a rich source of attack data. Honeypot data is one of the few ways by which new (zero-day) attacks can be detected. Honeypots also can serve as decoys that imitate important systems like those of command-and-control networks.

Honeypots are often used in groups called "honeynets" to provide plenty of targets for attacks and to study how attacks spread from computer to computer. Software for building honeynets is provided by the Honeynet Project (a consortium of researchers that provides open-source software) as well as some commercial vendors. Honeypot and honeynets can be "low-interaction" (simulating just the first steps of network protocols (Cohen & Koike, 2004)) or "high-interaction" (permitting logins and most resources of their systems, like Sebek (The Honeynet Project, 2004)). Low-interaction honeypots can fool attackers into thinking there are many good targets by simulating many Internet addresses and many vulnerable-looking services, as "decoys." For instance, low-interaction honeypots could implement a decoy military command-and-control network, so adversaries would attack it, rather than the real network. Low-interaction honeypots, like HoneyD, provide little risk to the deployer but are not very deceptive, since they usually must be preprogrammed with a limited set of responses. High-interaction honeypots, like Sebek, are more work to install and entail more risk of propagating an attack (since countermeasures cannot be perfect), but will fool more attackers and provide more useful data. A safer form of high-interaction honeypot is a "sandbox," a simulated environment that appears to be a real computer environment; it is important for forensics on malicious code.

Counterdeception and Counter-Counterdeception for Honeypots

Deception is necessary for honeypots because attackers do not want their activities recorded: This could permit legal action against them as well as learning of their tricks. So some attackers search for evidence of honeypots on systems into which they trespass; this is a form of counterdeception (McCarty, 2003). Analogously to intrusion-detection systems for cyberspace (Proctor, 2001), this counterdeception can either look for statistical anomalies or for features or "signatures" that suggest a honeypot. Anomalies can be found in statistics on the types, sizes, and dates of files and directories. For instance, a system with no e-mail files is suspicious. Counter-counterdeception in designing good honeypots then requires ensuring realistic statistics on the honeypot. A tool that calculates statistical metrics on typical computer systems is useful (Rowe, 2006). One good way to build a honeypot is to copy its file system from a typical real computer system. However, exactly identical file systems are suspicious so it is important to make at least random differences among honeypots.

Honeypot signatures can be found in main memory, secondary storage, and network packets (Holz &