# If you go down to the Internet today – Deceptive Honeypots

Suen Yek and Craig Valli

School of Computer and Information Science
Edith Cowan University
E-mail: syek@student.ecu.edu.au

School of Computer and Information Science
Edith Cowan University
E-mail: c.valli@ecu.edu.au

**ABSTRACT**
*This is preliminary research into the effectiveness of deceptive defensive measures in particular honeypots that use deceit as a primary defensive and offensive mechanism. Initial research has been conducted using the Deception Tool Kit and its ability to fool commonly available network scanning tools such as Nessus and Nmap The preliminary research indicates that these deceptive tools have a place in modern network defence architecture.*

**Keywords** *deception, network, security*

## INTRODUCTION

Attacking trends over the last 5 years have shown Internet connections to be the increasingly cited point of attack (Power, 2002). This challenges the prior conception that most attacks are internal. While inside attacks still show significant numbers, the growth in reported out-sourced attacks show up to 60% on WWW/Company sites (Power, 2002).

Statistics show 90% of respondents detected computer security breaches with financial losses within the last 12 months. Furthermore up to 40% detected Denial of Service (DoS) attacks (Power, 2002). These attacks indicate the mounting concern of defacing company reputation combined with theft of proprietary information, and financial fraud.

The goal of this project was to determine the ability for trapping and analysing the results of potentially dangerous attacks on a server when using a honeypot as the prime forensic gathering tool. For this purpose, a honeypot will then be defined as a 'resource whose value is in being attacked or compromised' (Spitzner, 2002).

These experiments were carried out in a private and secluded network of eight workstations and a server within the University. The victim machine was running Linux Redhat 7.2 operating system (OS) and had the DTK installed. The attack PC was primarily a Linux Redhat 7.2 machine, a Windows2000 PC was used to confirm results from the Linux tests with Nmap as a cross platform tool. The DTK itself used several deceptive operating systems with decoy port addresses and outputs to re-direct probes to the DTK. Information regarding the attacks were recorded via the logging features of the DTK and the conventional syslogd daemon. These results were then analysed to deduce the level of effectiveness of the honeypot in a real life situation.

### Why use a deceptive honeypot?

The DTK has the ability to deceptively mimic the following operating systems: Windows NT, Linux, HP-UX, SCO Unix, SGI, IBM AIX, Sun Solaris, SunOS and Ultrix. The deception toolkit has been used as the primary architecture of the honeypot. Firstly it was designed to be used as a defensive tool that systems administrators could use to defend systems. The DTK mimics industry standard servers and the services they provide by listening for inputs and redirecting traffic to customisable PERL

script files. These script files then respond as an installed server or daemon should when sent commands that are legitimate or otherwise, selected

By deploying bogus services the machine will appear to contain seemingly numerous useful ports, and it will output responses that are intended to appear typical of a functioning server. While doing this, the DTK will record the actions of the intruder through log files that can then be analysed to determine the modus operandi of the attacker. The deceptive honeypot is intended to extend the time detection window as the attacker is drawn into probing services that are digital chameleons that have no real payload or substance.

The data collected from honeypot testing is 'normally of high value' (Spitzner, 2002). This is because information extracted from the analysis can be easily collected, organised, and documented. The high value information includes network activity and movements of the attacker, once in the system. Additionally, honeypots only capture information that is targeted to it. Therefore there is no overwhelming bandwidth or activity to overlook network progress by dropping packets, and potential attacks (Spitzner, 2002). Consequently there is a more efficient use of resources to provide manageable amounts of useful data for providing attack intelligence to the defender.

The honeypot was designed to act as a fully functional mail server installed within a typical online business running SSH, SMTP and POP3 services. For optimal operation in a real life situation, the honeypot would be required to be set up in an independent location away from the legitimate servers or within a tightly controlled and protected DMZ (Demilitarised Zone).

Design of the honeypot involves allocating inactive ports to a potentially viable host. This is intended to deceive the attacker into thinking they can receive valuable information from scanning port traffic and determining where that port connects to and identifying flaws in the movement of traffic (McClure, Scambray, & Kurtz, 2002).

The intended function of the honeypot is to confuse and disorientate the attacker by falsely directing them through bogus host lines that may or may not provide information that appears to be informative or even important. As there will seem to be numerous available ports, scanning will take considerably more time, depending on the configuration of the honeypot, and consequential disorientation may result in one of two ways. Firstly, the attacker will be bored, confused or angered and will cease hacking attempts. Secondly, the attacker will believe they have hacked into the system server, and will believe they have received valuable information.

In either case, the honeypot will have achieved its desired purpose in keeping record of the actions taken by the attacker unknowingly, and their consequent attacking strategies.

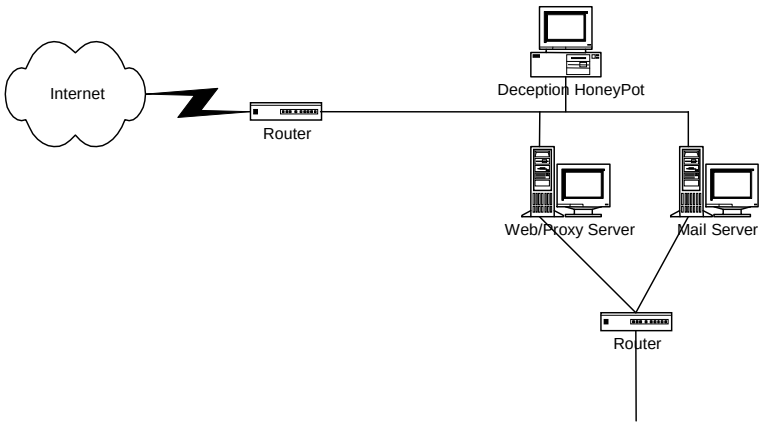


**Diagram 1 - Honeypot deployment**

Diagram 1 depicts a situation where a potential intruder enters through an Internet connection to the company router. The router has a connection to all network stations, including the honeypot. It then automatically directs the intruder to the honeypot, and away from genuine company assets. Similarly, an internal router directs traffic from within the company network without interference to the honeypot.

## METHODOLOGY

There are various types of system testing and penetrating tools freely available to download from the World Wide Web. Two popular attacking softwares chosen were Nessus and Nmap. Both of these tools have won various industry accolades for software innovation and best of breed.

Nessus is a popular network security scanning and auditing tool (Insecure, 2002). Nessus checks for vulnerable systems by detecting all the ports running any given service and then probes and tests their security against known vulnerabilities. Nessus uses a server/daemon: nessusd, and a client nessus. Nessusd monitors the attacks and locates the security holes, then reports them to the Nessus client. The client then interfaces with the user and displays the results.

The results show information on which ports were scanned and corresponding responses. The client output references to possible security holes and exposure. More importantly, for the scope of this research, buffer overflows on ports suggest points of entry and manipulation to malicious users to initiate a DoS attack as a subsequent vulnerability.

Nmap (Network Mapper) is a freely available utility used for network exploration or security auditing. By using a technique known as OS fingerprinting (Fyodor, 1998) which examines returned IP packets received from the host Nmap is able to determine hosts available on the network, ports used, any packet filters or firewalls in use and what operating systems and versions are in use.

The attacks were performed through specifying ranges of ports to scan. These may be upon the assumption that the potential outside intruder has already performed some form of systematic fingerprinting, which is a tactic used to obtain company profiles of domain names, network blocks, and individual IP addresses connected to the Internet (McClure et al, 2001). Alternatively, the assumption also can be that the intruder may be internal, where the IP addresses are already known, or easily accessible. Though not the highest, this is also a common source for attacks (Power, 2002).

Once IP address ranges are known, the intruder will perform port scanning in order to determine live hosts. This is often time consuming on the attackers part and is not entirely conclusive or accurate (McClure et al, 2002). The information that a potential intruder will receive can range from complete disclosure of the system's makeup including operating system (OS), network configuration and loaded services. This then allows the attacker to identifying related OS and service vulnerabilities. As many security vulnerabilities and exploits are dependent on the OS version an attacker can easily adjust their code to attack those weaknesses (Fyodor, 1998).

Through Nessus, brute force attacks and covert methods were used. The DTK was implemented on the victim host. Nessus was chosen as the software to initiate attacks against the victim host running the DTK utilising the brute force modes of Nessus. The thoroughness of the results of the attacks can then be compared to the actual logged information taken from the DTK log files and the standard syslogd facilities on the Linux system. Thus the honeypot DTK showed its level of effectiveness in distracting the attacker from real port addresses and its potential to prevent hazardous damage.

Limitations on the research are that the seclusion of the experimental network does not connect to the Internet and World Wide Web. This honeypot was designed to act as a fully functional mail server installed and would operate with SSH, SMTP and POP3 services.

## TESTING AND EVALUATION PLAN

Nessus was used to brute force the DTK in each of its deceptive OS's. The probed ports were 1 - 1024, 12345, 1246, 2049, 5999 - 8000, 10000 – 28000. The Maximum number of threads was set to 8 and TCP connect scans were used to probe the ports on the victim host.

When the scan was complete a report was generated by Nessus with any detected security warnings and associated notes. A sample follows for the SMTP service when using SGI deception.

> **Warning found on port smtp (25/tcp)**
>
> The remote STMP server seems to allow remote users to send mail anonymously by providing a too long argument to the HELO command (more than 1024 chars). This problem may allow bad guys to send hate mail, or threatening mail using your server and keep their anonymity.
>
> Risk factor : Low.
>
> Solution : If you are using sendmail, upgrade to version 8.9.x. If you do not run sendmail, contact
> your vendor.
> CVE : CAN-1999-0098
>
> **Information found on port smtp (25/tcp)**
>
> Remote SMTP server banner :
> netsec.ecu SGI ESMTP Sendmail 8.1.2/8.1.3

## ANALYSIS OF RESULTS

When all scans were complete on each deceptive OS, the log files that the DTK generated for each OS were then imported into Excel spreadsheets for further viewing and analysis. Filtered data was retrieved and sorted into the spreadsheet. It was then evaluated by simply counting and recording buffer overflows on each of the probed OS's.

| Port | AIX | SGI | SUN | ULTRIX |
|---|---|---|---|---|
| **19** | 5 | 2 | | 2 |
| **25** | 94 | 31 | 2 | 32 |
| **110** | 75 | 24 | 5 | 26 |
| **365** | 5 | 2 | | 2 |
| **893** | 5 | 2 | | 2 |
| **2049** | 6 | 2 | | 2 |
| **5999** | 6 | 2 | | 2 |
| **6001** | 6 | 2 | | 2 |
| **8000** | 6 | 2 | | 2 |
| **10000** | 1 | | | |
| **12345** | 4 | 2 | | 2 |
| **12346** | 4 | 2 | | 2 |

**Table 1 – Deceptive Buffer Overflows by Port/Service**

Many security holes publicised are due to buffer overflows as a form of attack on company servers (Graham, 2000). Therefore it is a common problem faced and is a notable response from the Nessus reports. The number of buffer overflow indicates the number of times the DTK was able to output a

red herring to intruders. Where an overflow of data is normally generated, there is a high likelihood that the program will crash or give an intruder root or high level access or privilege to a system.

Nessus believed it detected the following problems with the various deceptive operating systems

| | Security | | | Rating of Problems | | | |
|---|---|---|---|---|---|---|---|
| | **Holes** | **Warn** | **Notes** | **Serious** | **High** | **Med** | **Low** |
| **LINUX** | 1 | 5 | 3 | 0 | 50 | 50 | 0 |
| **NT** | 0 | 7 | 5 | 0 | 0 | 0 | 100 |
| **SOLARIS** | 0 | 6 | 4 | 0 | 0 | 0 | 100 |
| **HPUX** | 0 | 8 | 5 | 0 | 0 | 12 | 88 |
| **SUNOS** | 0 | 7 | 5 | 0 | 0 | 0 | 100 |
| **AIX** | 0 | 7 | 5 | 0 | 0 | 0 | 100 |
| **SGI** | 0 | 7 | 5 | 0 | 0 | 0 | 100 |
| **Ultrix** | 0 | 7 | 5 | 0 | 0 | 0 | 100 |
| **SCO** | 0 | 6 | 5 | 0 | 0 | 0 | 100 |

**Table 2 - Nessus Results No Dangerous Plugins Used**

| | Security | | | Rating of Problems | | | |
|---|---|---|---|---|---|---|---|
| | **Holes** | **Warn** | **Notes** | **Serious** | **High** | **Med** | **Low** |
| **LINUX** | 1 | 5 | 3 | 0 | 50 | 50 | |
| **NT** | 0 | 8 | 5 | 0 | 0 | 0 | 100 |
| **SOLARIS** | 0 | 6 | 4 | 0 | 0 | 0 | 100 |
| **HPUX** | 1 | 8 | 5 | 0 | 0 | 12 | 88 |
| **SUNOS** | 0 | 8 | 5 | 0 | 0 | 0 | 100 |
| **AIX** | 0 | 8 | 5 | 0 | 0 | 0 | 100 |
| **SGI** | 1 | 8 | 5 | 0 | 0 | 12 | 88 |
| **Ultrix** | 1 | 8 | 5 | 0 | 0 | 12 | 88 |
| **SCO** | 0 | 7 | 5 | 0 | 0 | 0 | 100 |

**Table 3 - Nessus Results Dangerous Plugins Used**

A naïve hacker or script kiddie would typically rely heavily on tools such as Nessus to provide them with potential targets that they could compromise (Conry-Murray, 2001). This reliance on pre-compiled tools where they do not have to understand or be able to manually execute the attack is a hole in their offensive strategy. This enables the defenders to extend the detection window for the attacker as they are literally shadow boxing in a deceptive honeypot while leaving forensic trails of their activities.

Although the detected problems were rated as low they still leave an opportunity for a hacker to attempt to attack the system.

The SMTP service was the service that provided the most deceptive information and demonstrated the highest level of buffer overflows to a would be attacker. All of the deceptive OS implementation provided faked remote banners that took the form of netsec.ecu (Deceptive OS) SMTP Sendmail 8.1.2/8.1.3. In hacking guides (Anonymous, 2002; Fadia, 2002) commonly available on the Internet that target the SMTP service and in particular the Sendmail program, this information is used to determine what sort of attack the attacker should deploy to penetrate or dupe the system. This is some of the initial intelligence gathering that an attacker would undertake. Based on that knowledge they would then attempt various exploits on the system. Nessus believed it perpetrated buffer overflows that resulted in denial of service or allowed the successful anonymous relay of mail. It also further

believed that the mail server was an open mail relay (Finlay et al, 2001; Rosenthal, 2002) which allows malicious attackers to pass mail through the server to other persons.

The SSH service in deception mode mimicked buffer overflows and core dumps whereby the attacker would once again believe they would have performed a successful denial of service on the SSH daemon. The POP3 service also gave away banner information which a hacker could use in the same manner as the SMTP banner to search for vulnerabilities.

All of the deceptive OS's demonstrated to Nessus vulnerabilities that simply did not exist in any real form on the victim system. This further confirms the veracity of claims (Cohen, 1998) 'The net effect is that attack tools that automatically scan for known vulnerabilities find what appear to be large volumes of vulnerabilities. When the attacker tries to interpret the results of automated scans, there is not enough information to tell which of the detected vulnerabilities are real, and the number of detected vulnerabilities is very high and dominated by deceptions.'

One of the problems encountered was the DTK's inability to counteract Nmap OS fingerprinting techniques. The DTK returned consistently inconclusive results guessing that the OS was Standard: Solaris 2.x, Linux 2.1.???, Linux 2.2, MacOS. This gives the attacker a choice of four OS's to choose from or would potentially be used as fingerprint of a DTK by a wily attacker. This would pose a potential weakness when multi-homing sites on the one system using network aliasing.

## CONCLUSION

The use of deception for defence has been around since the dawn of time and will be here from some time to come. Its place in a network defensive strategy is still relatively unclear. The use of deception based systems such as the DTK has the ability to fool many of the common scanners used by naïve or inexperienced attackers. This then leaves the naïve hacker at the mercy of their ignorance in successfully attacking the real system that in turn provides the defender time to instigate countermeasures to prevent further attack, or redirect further attempted incursions.

The DTK as tested provided extensive forensic data in its log files and to the syslogd functions on the attacked system. This forensic data would aid greatly in the investigation of an attempted break in.

The use of deceptive honeypots has weaknesses that need examination and further resolution if they are to be effective as a defensive mechanism. With the advent of multi-homing on one system/interface the ability to deceptively portray that interface as multiple systems/interfaces is an important extension of a deceptive honeypot to cope with modern networking technologies. Whether a more intricate deception that is more detailed and descriptive will aid in increasing the deception needs further investigating.

## References

Cohen, F. (1998). A note on the role of deception in information protection. *Computers & Security*, **17**(6), pp.483-506.

Conry-Murray, A. (2001). Network security's not-so-secret ingredients. *Network Magazine*, **16**(8).

Fadia, A. (2002 – last update), "Sendmail and Beyond: Kewl Tips and Tricks", (*Hacking Truths*), Available: http://hackingtruths.box.sk/smtp.htm (Accessed: 2002, June 1).

Finlay, I., King, B., & Hernan, S. (2001). CERT® Incident Note IN-2001-02 - Open mail relays used to deliver "Hybris Worm": CERT.

Fyodor. (1998). "Remote OS detection via TCP/IP stack fingerprinting", (*Insecure: Nmap*), Available: http://www.insecure.org/nmap/nmap-fingerprinting-article.txt (Accessed: 2002, May10).

Graham, R (2000, March 21 – last update). “FAQ: Network Intrusion Detection Systems” (*RobertGraham.com Infosec pubs*). Available: http://www.robertgraham.com/pubs/network-intrusion-detection.html#11 (Accessed: 2002, May 22).

“Nmap – Introduction”, (2002, August 10 – last update), (*Insecure.org*), Available: http://www.insecure.org/nmap/ (Accessed: 2002, May 10).

McClure, S., Scambray, J. & Kurtz. G. (2001). *Hacking Exposed: Network Security Secrets and Solutions* (3rd ed.): McGraw Hill, USA.

Power, R. (2002).Computer Security Issues and Trends. *CSI/FBI Computer Crime and Security Survey* **8**(1).

Raven. (2002), “Sendmail”, (*Hacking*), Available: http://www.astalavista.com/archive/hacking/sendmail.htm (Accessed: 2002, June 1).

Rosenthal, C. (2001, April 23 – last update), “What is Third-Party Mail Relay?”, (*Maps: Transport Security Initiative*), Available: http://mail-abuse.org/tsi/ar-what.html (Accessed: 2002, June 1).

Spitzner, L. (2002, May 17 – last update), “Honeypots: Definitions and Value of Honeypots”, (*Lance’s Security Papers*), Available: http://www.enteract.com/~lspitz/honeypot.html (Accessed: 2002, April 14).