An Analysis of the Deflect

In computer terminology, a **Deflect** is a trap set to detect or in some manner counteract attempts at unauthorized use of information systems. Generally it consists of a computer, data, or a network site that appears to be part of a network, but is actually isolated and monitored, and which seems to contain information or a resource of value to attackers.

1.1 Function

A Deflect is valuable as a surveillance and early-warning tool. While it is often a computer, a Deflect can take other forms, such as files or data records, or even unused IP address space. A Deflect that masquerades as an open proxy to monitor and record those using the system is known as a "sugarcane". Counteract should have no production value, and hence should not see any legitimate traffic or activity. Whatever they capture is therefore malicious or unauthorized. One practical application of this is the spamtrap - a Deflect that thwarts spam by masquerading as a type of system abused by spammers. These counteract categorize trapped material 100% accurately: it is all illicit. Counteract can carry risks to a network, and must be handled with care. If they are not properly walled off, an attacker can use them to break into a system.

**Victim hosts** are an active network counter-intrusion tool. These computers run special software, designed to appear to an intruder as being important and worth looking into. In reality, these programs are dummies, and their patterns are constructed specifically to foster interest in attackers. The software installed on, and run by, victim hosts is dual purpose. First, these dummy programs keep a network intruder occupied looking for valuable information where none exists, effectively convincing an intruder to isolate themselves in what is truly an unimportant part of the network. This decoy strategy is designed to keep an intruder from getting bored and heading into truly security-critical systems. The second part of the victim host strategy is intelligence gathering. Once an intruder has broken into the victim host, the machine or a network administrator can examine the intrusion methods used by the intruder. This
intelligence can be used to build specific countermeasures to intrusion techniques, making truly important systems on the network less vulnerable to intrusion.

1.2 Types

Counteract can be classified based on their deployment and based on their level of involvement. Based on the deployment, counteract may be classified as

1.2.1 Production Counteract

Production counteract are easy to use, capture only limited information, and are used primarily by companies or corporations; Production counteract are placed inside the production network with other production servers by an organization to improve their overall state of security. Normally, production counteract are low-interaction counteract, which are easier to deploy. They give less information about the attacks or attackers than research counteract do. The purpose of a production Deflect is to help mitigate risk in an organization. The Deflect adds value to the security measures of an organization.

1.2.2 Research Counteract

Research counteract are run by a volunteer, non-profit research organization or an educational institution to gather information about the motives and tactics of the Blackhat community targeting different networks. These counteract do not add direct value to a specific organization; instead, they are used to research the threats organizations face and to learn how to better protect against those threats. This information is then used to protect against those threats. Research counteract are complex to deploy and maintain, capture extensive information, and are used primarily by research, military, or government organizations. Based on the design criteria, counteract can be classified into three categories as

1.2.3 Pure Counteract

Pure Deflect is a fully fledged production system. The activities of the attacker is monitored using a casual tap has been installed on the counteract link to the network. No other
software's are needed to be installed. Even though a pure Deflect is useful, stealthiness of the defense mechanisms can be ensured by a more controlled mechanism.

### 1.2.4 High Interaction Counteract

High interaction counteract imitate the activities of the real systems that host a varieties of services and, therefore, an attacker may be allowed a lot of services to waste his time. According to recent researches in high interaction Deflect technology, by employing virtual machines, multiple counteract can be hosted on single physical machine. Therefore, even if the Deflect is compromised, there is chance for quicker recovery. In general, high interaction counteract provide more security by being difficult to detect but, on the negative side, are highly expensive to maintain. If virtual machines are not available, each Deflect need to maintain for each physical computer, which can be exorbitantly expensive. Example: Honeynet.

### 1.2.5 Low Interaction Counteract

Low interaction Deflect is based on the services that the attacker normally request for. These services are simulated by this classification of Deflect. There are many positives with the requirement of only few services by the attackers: ease of hosting multiple virtual machines on one physical system as they consume relatively few resources, fast response time of the virtual systems and shorter code length reduces the complexity in the security of the virtual systems. Example: Honeyd.

### 1.2.6 Spam versions

Spammers abuse vulnerable resources such as open mail relays and open proxies. Some system administrators have created Deflect programs that masquerade as these abusable resources to discover spammer activity. There are several capabilities such counteract provide to these administrators and the existence of such fake abusable systems makes abuse more difficult or risky. Counteract can be a powerful countermeasure to abuse from those who rely on very high volume abuse (e.g., spammers).

These counteract can reveal the apparent IP address of the abuse and provide bulk spam capture (which enables operators to determine spammers' URLs and response
mechanisms). For open relay counteract, it is possible to determine the e-mail addresses ("dropboxes") spammers use as targets for their test messages, which are the tool they use to detect open relays. It is then simple to deceive the spammer: transmit any illicit relay e-mail received addressed to that dropbox e-mail address. That tells the spammer the Deflect is a genuine abusable open relay, and they often respond by sending large quantities of relay spam to that Deflect, which stops it. The apparent source may be another abused system—spammers and other abusers may use a chain of abused systems to make detection of the original starting point of the abuse traffic difficult.

This in itself is indicative of the power of counteract as anti-spam tools. In the early days of anti-spam counteract, spammers, with little concern for hiding their location, felt safe testing for vulnerabilities and sending spam directly from their own systems. Counteract made the abuse less easy and safe. Spam still flows through open relays, but the volume is much smaller than in 2001 to 2002. While most spam originates in the U.S., spammers hop through open relays across political boundaries to mask their origin. Deflect operators may use intercepted relay tests to recognize and thwart attempts to relay spam through their counteract. "Thwart" may mean "accept the relay spam but decline to deliver it".

Deflect operators may discover other details concerning the spam and the spammer by examining the captured spam messages. (However, open relay spam has declined significantly). Open relay counteract include Jackpot, written in Java, smtpot.py, written in Python, and spamhole, written in C. The Bubblegum Proxypot is an open proxy Deflect (or proxypot).

1.2.7 E-mail trap

An e-mail address that is not used for any other purpose than to receive spam can also be considered a spam Deflect. Compared with the term spamtrap, the term "Deflect" might better be reserved for systems and techniques used to detect or counter attacks and probes. Spam arrives at its destination "legitimately"—exactly as non-spam e-mail would arrive.

An amalgam of these techniques is Project Honey Pot. The distributed, open-source Project uses Deflect pages installed on websites around the world. These Deflect pages hand out
uniquely tagged spamtrap e-mail addresses. E-mail address harvesting and Spammers can then be tracked as they gather and subsequently send to these spamtrap e-mail addresses.

1.3 Database Deflect

Databases often get attacked by intruders using SQL Injection. Because such activities are not recognized by basic firewalls, companies often use database firewalls. Some of the available SQL database firewalls provide/support Deflect architectures to let the intruder run against a trap database while the web application still runs as usual.

1.3.1 Detection of Deflect

Just as counteract are weapons against spammers, Deflect detection systems are spammer-employed counter-weapons. As detection systems would likely use unique characteristics of specific counteract to identify them, a great deal of counteract in use makes the set of unique characteristics larger and more daunting to those seeking to detect and thereby identify them. This is an unusual circumstance in software: a situation in which "versionitis" (a large number of versions of the same software, all differing slightly from each other) can be beneficial. There's also an advantage in having some easy-to-detect counteract deployed. Fred Cohen, the inventor of the Deception Toolkit, even argues that every system running his Deflect should have a deception port that adversaries can use to detect the Deflect. Cohen believes that this might deter adversaries.

1.3.2 Honeynets

Two or more counteract on a network form a honeynet. Typically, a honeynet is used for monitoring a larger and/or more diverse network in which one Deflect may not be sufficient. Honeynets and counteract are usually implemented as parts of larger network intrusion detection systems. A honeyfarm is a centralized collection of counteract and analysis tools. The concept of the honeynet first began in 1999 when Lance Spitzner, founder of the Honeynet Project, published the paper "To Build a Deflect":
"A honeynet is a network of high interaction counteract that simulates a production network and configured such that all activity is monitored, recorded and in a degree, discreetly regulated."

1.4 Client Deflect

Counteract are security devices whose value lie in being probed and compromised. Traditional counteract are servers (or devices that expose server services) that wait passively to be attacked. **Client Counteract** is active security devices in search of malicious servers that attack clients. The client Deflect poses as a client and interacts with the server to examine whether an attack has occurred. Often the focus of client counteract is on web browsers, but any client that interacts with servers can be part of a client Deflect (for example ftp, ssh, email, etc.). There are several terms that are used to describe client counteract. Besides client Deflect, which is the generic classification, honeyclient is the other term that is generally used and accepted. However, there is a subtlety here, as "honeyclient" is actually a homograph that could also refer to the first open source client Deflect implementation (see below), although this should be clear from the context.

1.4.1 Architecture of Client Deflects

A client Deflect is composed of three components. The first component, a queuer, is responsible for creating a list of servers for the client to visit. This list can be created, for example, through crawling. The second component is the client itself, which is able to make a requests to servers identified by the queuer.

After the interaction with the server has taken place, the third component, an analysis engine, is responsible for determining whether an attack has taken place on the client Deflect. In addition to these components, client counteract are usually equipped with some sort of containment strategy to prevent successful attacks from spreading beyond the client Deflect. This is usually achieved through the use of firewalls and virtual machine sandboxes.

Analogous to traditional server counteract, client counteract are mainly classified by their interaction level: high or low; which denotes the level of functional interaction the server
can utilize on the client Deflect. In addition to this there are also newly hybrid approaches which denotes the usage of both high and low interaction detection techniques.

1.4.2 High Interaction of Client Deflects

High interaction client counteract are fully functional systems comparable to real systems with real clients. As such, no functional limitations (besides the containment strategy) exist on high interaction client counteract. Attacks on high interaction client counteract are detected via inspection of the state of the system after a server has been interacted with. The detection of changes to the client Deflect may indicate the occurrence of an attack against that has exploited a vulnerability of the client. An example of such a change is the presence of a new or altered file.

High interaction client counteract are very effective at detecting unknown attacks on clients. However, the tradeoff for this accuracy is a performance hit from the amount of system state that has to be monitored to make an attack assessment. Also, this detection mechanism is prone to various forms of evasion by the exploit. For example, an attack could delay the exploit from immediately triggering (time bombs) or could trigger upon a particular set of conditions or actions (logic bombs). Since no immediate, detectable state change occurred, the client Deflect is likely to incorrectly classify the server as safe even though it did successfully perform its attack on the client. Finally, if the client counteract are running in virtual machines, then an exploit may try to detect the presence of the virtual environment and cease from triggering or behave differently.

1.4.2.1 Capture-HPC

Capture is a high interaction client Deflect developed by researchers at Victoria University of Wellington, NZ. Capture differs from existing client counteract in various ways. First, it is designed to be fast. State changes are being detected using an event based model allowing to react to state changes as they occur. Second, Capture is designed to be scalable. A central Capture server is able to control numerous clients across a network. Third, Capture is supposed to be a framework that allows to utilize different clients. The initial version of Capture supports Internet Explorer, but the current version supports all major browsers
(Internet Explorer, Firefox, Opera, Safari) as well as other HTTP aware client applications, such as office applications and media players.

1.4.2.2  **HoneyClient**

HoneyClient is a web browser based (IE/FireFox) high interaction client Deflect designed by Kathy Wang in 2004 and subsequently developed at MITRE. It was the first open source client Deflect and is a mix of Perl, C++, and Ruby. HoneyClient is state-based and detects attacks on Windows clients by monitoring files, process events, and registry entries. It has integrated the Capture-HPC real-time integrity checker to perform this detection. HoneyClient also contains a crawler, so it can be seeded with a list of initial URLs from which to start and can then continue to traverse web sites in search of client-side malware.

1.4.2.3  **HoneyMonkey**

HoneyMonkey is a web browser based (IE) high interaction client Deflect implemented by Microsoft in 2005. It is not available for download. HoneyMonkey is state based and detects attacks on clients by monitoring files, registry, and processes. A unique characteristic of HoneyMonkey is its layered approach to interacting with servers in order to identify zero-day exploits. HoneyMonkey initially crawls the web with a vulnerable configuration. Once an attack has been identified, the server is reexamined with a fully patched configuration. If the attack is still detected, one can conclude that the attack utilizes an exploit for which no patch has been publicly released yet and therefore is quite dangerous.

1.4.2.4  **Shelia**

Shelia is a high interaction client Deflect developed by Joan Robert Rocaspana at Vrije Universiteit Amsterdam. It integrates with an email reader and processes each email it receives (URLs & attachments). Depending on the type of URL or attachment received, it opens a different client application (e.g. browser, office application, etc.) It monitors whether executable instructions are executed in data area of memory (which would indicate a buffer overflow exploit has been triggered). With such an approach, SHELIA is not only able to detect exploits, but is able to actually ward off exploits from triggering.
1.4.2.5 UW Spycrawler

The Spycrawler developed at the University of Washington is yet another browser based (Mozilla) high interaction client Deflect developed by Moshchuk. in 2005. This client Deflect is not available for download. The Spycrawler is state based and detects attacks on clients by monitoring files, processes, registry, and browser crashes. Spycrawlers detection mechanism is event based. Further, it increases the passage of time of the virtual machine the Spycrawler is operating in to overcome (or rather reduce the impact) of time bombs.

1.4.2.6 Web Exploit Finder

WEF is an implementation of an automatic drive-by-download – detection in a virtualized environment, developed by Thomas Müller, Benjamin Mack and Mehmet Arziman, three students from the Hochschule der Medien (HdM), Stuttgart during the summer term in 2006. WEF can be used as an active HoneyNet with a complete virtualization architecture underneath for rollbacks of compromised virtualized machines.

1.4.3 Low Interaction Client Deflects

Low interaction client counteract differ from high interaction client counteract in that they do not utilize an entire real system, but rather use lightweight or simulated clients to interact with the server. (in the browser world, they are similar to web crawlers). Responses from servers are examined directly to assess whether an attack has taken place. This could be done, for example, by examining the response for the presence of malicious strings. Low interaction client counteract are easier to deploy and operate than high interaction client counteract and also perform better. However, they are likely to have a lower detection rate since attacks have to be known to the client Deflect in order for it to detect them; new attacks are likely to go unnoticed. They also suffer from the problem of evasion by exploits, which may be exacerbated due to their simplicity, thus making it easier for an exploit to detect the presence of the client Deflect.

1.4.3.1 HoneyC
HoneyC is a low interaction client Deflect developed at Victoria University of Wellington by Christian Seifert in 2006. HoneyC is a platform independent open source framework written in Ruby. It currently concentrates driving a web browser simulator to interact with servers. Malicious servers are detected by statically examining the web server’s response for malicious strings through the usage of Snort signatures.

1.4.3.2 Monkey-Spider

Monkey-Spider is a low-interaction client Deflect initially developed at the University of Mannheim by Ali Ikinci. Monkey-Spider is a crawler based client Deflect initially utilizing anti-virus solutions to detect malware. It is claimed to be fast and expandable with other detection mechanisms. The work has started as a diploma thesis and is continued and released as Free Software under the GPL.

1.4.3.3 PhoneyC

PhoneyC is a low-interaction client developed by Jose Nazario. PhoneyC mimics legitimate web browsers and can understand dynamic content by de-obfuscating malicious content for detection. Furthermore, PhoneyC emulates specific vulnerabilities to pinpoint the attack vector. PhoneyC is a modular framework that enables the study of malicious HTTP pages and understands modern vulnerabilities and attacker techniques.

1.4.3.4 SpyBye

SpyBye is a low interaction client Deflect developed by Niels Provos. SpyBye allows a web master to determine whether a web site is malicious by a set of heuristics and scanning of content against the ClamAV engine.

1.4.4 Hybrid Client Counteract

Hybrid client counteract combine both low and high interaction client counteract to gain from the advantages of both approaches.

1.4.4.1 HoneySpider
The HoneySpider network is a hybrid client Deflect developed as a joint venture between NASK/CERT Polska, GOVCERT.NL and SURFnet. The projects goal is to develop a complete client Deflect system, based on existing client Deflect solutions and a crawler specially for the bulk processing of URLs.