17-18 December 2012, ICRTITCS – 2012, TCSC, Mumbai, India
Proceedings published in International Journal of Computer Applications® (IJCA) (0975 – 8887)

Honeypot- A Tool to Trap Website Hackers

Juhi Danani
M.Sc(Computer Science)
Thakur College Of Science & Commerce

Jinal Jani
Assistant Professor
Computer Science Department
Thakur College of Science and Commerce

ABSTRACT
Honeypot is a closely monitored decoy that is employed in a network to study the trail of hackers and to alert network administrators of a possible intrusion. Honeynet is a methods for detection/protection/defense. Honeynet is an additional layer of security. Using Honeypot provides a cost-effective solution to increase the security posture of an organization. Even though it is not a panacea for security breaches, it is useful as a tool for network forensics and intrusion detection. Data Capture and Data Control are the properties of honeynet. Nowadays, they are also being extensively used by the research community to study issues in network security, such as Internet worms, spam control, DoS attacks, etc. In this paper, we advocate the use of honeypots as an effective educational tool to study issues in network security. We support this claim by demonstrating a set of projects that we have carried out in a Websites, which we have deployed specifically for running various web applications' under supervision . Primary intent of honeypot is to log and capture effects and activities of the threat.

Keywords
Security, tools, installation problem, framework, designing projects, Web application

1. INTRODUCTION
Global communication is getting more important every day. At the same time, computer crimes increasing. Counter measures are developed to detect or prevent attacks-most of these measures are based on known facts, known attack patterns. As in the military, it is important to know, who your enemy is, what kind of strategy he uses, what tools he utilizes and what he is aiming for. Gathering this kind of information is not easy but important. By knowing attack strategies, counter measures can be improved and vulnerabilities can be fixed. To gather as much information as possible is one main goal of honeypot. A honeypot is primarily an instrument for the information gathering and learning. Its primary purpose is not to be ambush for the blackhat community to catch them in action and to press charges against them. The lies on silent collection of as much information as possible about their attack patterns, used programs, purpose of attack and blackhat community itself. All this information is used to learn more about the blackhat proceedings and motives as well as their technical knowledge and abilities. This is just primary purpose if honeypot. There are a lot of other possibilities for a honeypot-divert hackers form productive systems for catch a hacker while conducting an attack are just two possible examples. Honeypots are not the perfect solution for solving or preventing computer crimes. Honeypots are hard to maintain and they need the good knowledge about the operating systems and network security. In the right hands honeypot is effective tool for the information gathering. In the wrong, inexperienced hands, a honeypot can become another infiltrated machine and an instrument for the black hat community.

1.1 HoneyPot BASICS
A honeypot is a resource whose value is being in attacked and compromised. This means, that a honeypot is expected to get probed, attacked and potentially exploited.

Honeypot do not fix anything. They provide us additional, valuable information. A honeypot is a resource, which pretends to be real target. A honeypot is expected to be attacked or compromised. The main goals are the distraction of an attacker and the gain of the information about the attack and the attacker.

There are two categories of honeypots.

- Production honeypots
- Research honeypots

A production honeypot is used to help migrate risk in an organization while the second category, is meant to gather as much information as possible. These honeypots do not add any security value to an organization, but they can help to understand the blackhat community and their attacks as well as to build some better defenses against security threats. A properly constructed honeypot is put on a network, which closely monitors the traffic to and from the honeypot. This data can be used for a variety of purposes.

- **Forensics**: analyzing new attacks and exploits
- **Trend analysis**: look for changes over time of types of attacks, techniques, etc
- **Identification**: track the bad people back to their home machines to figure out who they are.
- **Sociology**: learn about the bad guys as a group by snooping on email, IRC traffic, etc which happens to traverse the honeypot.

In general every traffic from and to a honeypot is unauthorized activity. All the data that is collected by a honeypot is therefore interested data. Data collected by the honeypot is of high value, and can lead to better understanding and knowledge which in turn can help to increase overall network security. One can also argue that a honeypot can be used for prevention because it can deter attackers from attacking other systems by occupying them long enough and bind their resources.

2. PROMBEM STATEMENT

There are kind of attacks on the websites or portals, because of which introduces new vulnerabilities into it.

Cross-site scripting (XSS) is a type of computer security vulnerability typically found in web applications that enables malicious attackers to inject client-side script into web pages viewed by other users. An exploited cross-site scripting vulnerability can be used by attackers to bypass access controls such as the same origin policy. Their impact may range from a petty nuisance to a significant security risk, depending on the sensitivity of the data handled by the vulnerable site, and the nature of any security mitigations implemented by the site's owner.

SQL injection is a code injection technique that exploits a security vulnerability occurring in the database layer of an application. The vulnerability is present when user input is either incorrectly filtered for string literal escape characters embedded in SQL statements or user input is not strongly typed and thereby unexpectedly executed. It is an instance of a more general class of vulnerabilities that can occur whenever one programming or scripting language is embedded inside another. SQL injection attacks are also known as SQL insertion attacks.

Remote File Inclusion (RFI) is a type of vulnerability most often found on websites, it allows an attacker to include a remote file usually through a script on the web server. The vulnerability occurs due to the use of user supplied input without proper validation. This can lead to something as minimal as outputting the contents of the file, but depending on the severity, to list a few it can lead to:

- Code execution on the web server
- Code execution on the client-side such as Javascript which can lead to other attacks such as cross site scripting (XSS).
- Denial of Service (DoS)
- Data Theft/Manipulation

CURL is a Client URL, a library created by Daniel Stenberg, is a predominantly command line based tool, which can be used to force parameters into a web request. The URL library was ported to PHP as an optional module and can be useful when attempting to gain reconnaissance information or unauthorized access to a designated URL. PHP supports libcurl which currently supports the http, https, ftp, gopher, telnet, dict, file, and ldap protocols. libcurl also supports HTTPS certificates, HTTP POST, HTTP PUT, FTP uploading (this can also be done with PHP's ftp extension), HTTP form based upload, proxies, cookies, and user + password authentication. CURL can be used in conjunction with PHP scripts for brute force attacks (including SQL injection table brute forcing), reconnaissance attacks, spoofing, and data theft.

3. OBJECTIVE

The system generated after implementing this paper acts as a service provider for Honeypot Security to various websites. It will be as a framework to implement honeypot which can be used by any organization to test their website applications / portals.

We plan to trace characteristics of hackers like.

- The browser they use.
- Their IP address from the IP header.
- The files accessed.
- The loopholes they discover.
- Various inputs that are used for various input fields
- Script Injection.

4. IMPLEMENTATION

Honeypots are closely monitored decoys that are employed in a network to study the trail of hackers and to alert network administrators of a possible intrusion. Theoretically, a Honeypot should see no traffic because it has no legitimate activity. This means any interaction with a Honeypot is most likely unauthorized or malicious activity.

The exact implementation of this project will be done using the following steps.

- IP tracing & HTTP packet analysis
- Honeytokens
- Honeypages
- Browser Defect Tracking
- Attacks Tracing [SQL Injection, Cross Side Scripting, etc].

4.1 IP tracing & HTTP packet analysis

We plan to inject certain scripts into the code of the web pages which will act as our Honeypot sniffer. These scripts could be JavaScript and SQL injections. These sniffers will then acquire the information and store it in our database. All unauthorized activities would then be tracked and stored in an administrative website for future analysis.

4.2 HoneyTokens

Honeytokens are fake records that are inserted in the database. These fake records are not expected to be used by normal users. If any of these honeytokens are used, they alert us of the database having been compromised. An example of honeytokens is fake username/passwords in the user database. These users do not exist in the real world, and hence are not expected to be logging in to the application. If the application sees these credentials being used, it immediately recognizes that the user database has been compromised.

4.3 HoneyPages

These are obscure web pages sprinkled in the web site. They have no legitimate purpose, nay they are not even linked from any valid page. Normal users would never reach these pages. However, we drop hints about these pages by embedding their url as comments or hidden fields in valid pages. While normal users would never see this, an attacker who analyzes the source code, or a vulnerability scanner that spiders the site would see these and follow the link. When the page is accessed, it points us to the intruder.

o Browser Defect tracking

All browsers have various configurations and accessibility. Hackers usually attack a website through loopholes in the browser. We intend to track the loopholes used by the hacker and change the settings of the website.

o Attacks Tracing- SQL injection

SQL injection is a code injection technique that exploits a security vulnerability occurring in the database layer of an application. The vulnerability is present when user input is either incorrectly filtered for string literal escape characters embedded in SQL statements or user input is not strongly typed and thereby unexpectedly executed. It is an instance of a more general class of vulnerabilities that can occur whenever one programming or scripting language is embedded inside another. SQL injection attacks are also known as SQL insertion attacks.

Coding for defining a static list to trap the hackers.

```
<?php


/* List with all attack-patterns to
search for
Contains names and strings for each
attack to display
Supports checking for word-segments or
whole words ($strictWholeWordDef),
supports case-sensitive and non-sensitive
checking ($caseSensitivityDef)*/
 class attacklist {

 protected $num_of_elements = 36;
// Number of attack-patterns in array
/* detection-array structure: index 0
= pattern
index 1 = strictWholeWord   true
means: only accept pattern,
if it occurs as a single word,
seperated by space characteres
index 2 = caseSensitivity true means:
only accept pattern, if it has exact
case-sensitive spelling
*/
// main array containing the attack
patterns to look for
protected $detectArray = NULL;
protected $strictWholeWordDef =
false;  // Default index 1: false
protected $caseSensitivityDef =
false;  // Default index 2: false
// array containing the thumbnail-
strings with html to return
protected $imgArray = NULL;
// array containing the short string-
names of the attacks to return
protected $strArray = NULL;
private static $noDetectionStr = "no
attack found";
// constructor creates the arrays and
sets varialbes
function __construct()
{
  for ( $i = 0; $i < $this-
>num_of_elements; $i++ )
{
$this->detectArray[ $i ][ 1 ] =
$this->strictWholeWordDef;
$this->detectArray[ $i ][ 2 ] =
$this->caseSensitivityDef;
}                              //
array with detection signatures    -
MANDATORY

$this->detectArray[ 0 ][ 0 ] =
htmlentities( "<script>" );
$this->detectArray[ 1 ][ 0 ] =
htmlentities( "</script>");
$this->detectArray[ 2 ][ 0 ] = "../";
$this->detectArray[ 3 ][ 0 ] = "/..";
$this->detectArray[ 4 ][ 0 ] = "OR";
$this->detectArray[4][1] = true;
$this->detectArray[4][2] = true;
$this->detectArray[ 5 ][ 0 ] =
"select"; $this->detectArray[5][1] =
true;
$this->detectArray[ 6 ][ 0 ] =
"insert";
$this->detectArray[ 7 ][ 0 ] =
"union";
$this->detectArray[ 8 ][ 0 ] =
"delete";
$this->detectArray[ 9 ][ 0 ] =
"where";
$this->detectArray[9][1] = true;
$this->detectArray[ 10 ][ 0 ] =
"wget";
$this->detectArray[ 11 ][ 0 ] =
"curl";
$this->detectArray[ 12 ][ 0 ] =
"lynx";
$this->detectArray[ 13 ][ 0 ] =
"fetch";
$this->detectArray[ 14 ][ 0 ] = "lwp-
download";
$this->detectArray[ 15 ][ 0 ] =
"echo"; $this->detectArray[15][1] =
true;
$this->detectArray[ 16 ][ 0 ] =
"javascript";
$this->detectArray[ 17 ][ 0 ] =
"`id`";
$this->detectArray[17][2] = true;
$this->detectArray[ 18 ][ 0 ] =
"uname";
$this->detectArray[ 19 ][ 0 ] =
"who";
$this->detectArray[19][1] = true;
$this->detectArray[ 20 ][ 0 ] =
"ifconfig";
$this->detectArray[ 21 ][ 0 ] =
htmlentities( "'" );
$this->detectArray[21][2] = true;
```

```
$this->detectArray[ 22 ][ 0 ] =
htmlentities( "`" );
$this->detectArray[22][2] = true;
$this->detectArray[ 23 ][ 0 ] =
htmlentities( "´" );
$this->detectArray[23][2] = true;
$this->detectArray[ 24 ][ 0 ] =
"http://";
$this->detectArray[ 25 ][ 0 ] =
"https://";
$this->detectArray[ 26 ][ 0 ] =
"include";
$this->detectArray[ 27 ][ 0 ] =
"select%";  // select in general
causes too many false positives
$this->detectArray[ 28 ][ 0 ] =
"select/";
$this->detectArray[ 29 ][ 0 ] =
"select#";
$this->detectArray[ 30 ][ 0 ] =
"select*";
$this->detectArray[ 31 ][ 0 ] =
"select\\";
$this->detectArray[ 32 ][ 0 ] =
"trigger";
$this->detectArray[ 33 ][ 0 ] =
"OUTFILE";
$this->detectArray[ 34 ][ 0 ] =
"passthru";
$this->detectArray[34][2] = true;
$this->detectArray[ 35 ][ 0 ] =
"exec";
$this->detectArray[35][2] = true;

// array with thumbnails to show
- OPTIONAL

 // array with attack-names to show
(instead of pictures)
$this->strArray[ 0 ] = "XSS";
$this->strArray[ 1 ] = "XSS";
$this->strArray[ 2 ] = "DIR-Change";
$this->strArray[ 3 ] = "DIR-Change";
$this->strArray[ 4 ] = "SQL";
$this->strArray[ 5 ] = "SQL";
$this->strArray[ 6 ] = "SQL";
$this->strArray[ 7 ] = "SQL";
$this->strArray[ 8 ] = "SQL";
$this->strArray[ 9 ] = "SQL";

$this->strArray[ 10 ] = "WGET";
$this->strArray[ 11 ] = "CURL";
$this->strArray[ 12 ] = "LYNX";
$this->strArray[ 13 ] = "FETCH";
$this->strArray[ 14 ] = "LWP";
$this->strArray[ 15 ] = "DEFACE";
$this->strArray[ 16 ] = "XSS";
$this->strArray[ 17 ] = "INJECTION";
$this->strArray[ 18 ] = "INJECTION";
$this->strArray[ 19 ] = "INJECTION";
$this->strArray[ 20 ] = "INJECTION";
$this->strArray[ 21 ] = "INJECT";
$this->strArray[ 22 ] = "INJECT";
$this->strArray[ 23 ] = "INJECT";
$this->strArray[ 24 ] = "INCLUSION";
$this->strArray[ 25 ] = "INCLUSION";
$this->strArray[ 26 ] = "INCLUSION";
$this->strArray[ 27 ] = "SQL";
$this->strArray[ 28 ] = "SQL";
$this->strArray[ 29 ] = "SQL";
$this->strArray[ 30 ] = "SQL";
$this->strArray[ 31 ] = "SQL";
$this->strArray[ 32 ] = "SQL";
$this->strArray[ 33 ] = "SQL";
$this->strArray[ 34 ] = "INCLUSION";
$this->strArray[ 35 ] = "INCLUSION";
}
 // returns the main detection-array
or NULL if nothing has been set
 public function getDetectionArray()
{
       return $this->detectArray;
}
// returns the number of attack
patterns detected
 public function getElementNr()
{
      return $this->num_of_elements;
 }
/* returns short-name-strings for
specified $id if no short-name-string
was found, error msg is returned */

public function getStrVal( $id )
 {
   if ( isset( $this->strArray[ $id
]))
   return $this->strArray[ $id];
   else
         return "unknown id:".$id;
        }
/* returns default-string to display
when no attack was found   */ public
static function getNoDetectionStr()
 {
            return
self::$noDetectionStr;
 }
 } // end: class

?>
```

5. FEATURES AND WORKING

The Paper will help to make a system which will do the following activities.

- Automatically scans for known attacks.
- Detects SLQ-Injections, (Remote) File-Inclusions, Cross-Site Scripting (XSS), Download attempts for malicious files e.g. with WGET or CURL, Command-Injections, etc.
- Provides an overview mode which allows you to look and scan for new incidents quickly (semi-automatic mode).
- Supports detailed information about all data correlated with every access to the honeypot. This includes but is not limited to HTTP-GET, HTTP-POST and COOKIE data.
- Saves copies of malicious tools in a secured place for later analysis.
- Provides a geographical, IP-based mapping about the attack sources. The generated map shows the origin of the attacks and offers additional details for each location.
- Generates numerous statistics about all traffic recognized at the system.

6. ADVANTAGES

- Honeypots only collect attack or unauthorized activity, dramatically reducing the amount of data they collect. Organizations that may log thousands of alerts a day may only log a hundred alerts with honeypots. This makes the data honeypots collect much easier to manage and analyze.
- Honeypots dramatically reduce false alerts, as they only capture unauthorized activity.
- Honeypots can easily identify and capture new attacks never seen before.
- Honeypots require minimal resources,even on the largest of networks. This makes them an extremely cost effective solution.
- Honeypots can capture encrypted attacks.

7. LIMITATIONS

- Honeypots can introduce risk to your environment. As we discuss later, different honeypots have different levels of risk. Some introduce very little risk, while others give the attacker entire platforms from which to launch new attacks, Risk is variable, depending on how one builds and deploys the honeypot.
- Implementing this system on an existing website could cause legality issues. Hence we intend to make dummy websites to demonstrate how our application functions.

8. CONCLUSION

A honeypot is just a tool. There are a variety of honeypot options, each having different value to organizations Production honeypots help reduce risk in an organization. Research honeypots are different in that they are not used to protect a specific organization. Instead they are used as a research tool to study and identify the threats in the Internet community. Regardless of what type of honeypot you use, keep in mind the 'level of interaction'. This means that the more your honeypot can do and the more you can learn from it, the more risk that potentially exists. You will have to determine what is the best relationship of risk to capabilities that exist for your problems. However, honeypots may be a tool to help contribute to those best practices.

9. REFERENCES

[1] Lance Spitzner. Honeypots: Tracking Hackers. Addison-Wesley, Boston. 2002.

[2] Cli®ord Stoll. Stalking the Wily Hacker. Communications of the ACM. pp 484-

[3] Honeynet Research Alliance. Project Honeynet Website. Retrieved May 16th

[4] Computer Emergency Response Team. dtscpd Exploit Advisory. Ad-

[5] Niels Provos. Honeyd. Retrieved May 16th 2003 from the World Wide

[6] http://www.citi.umich.edu/u/provos/honeyd/

[7] Dug Song and Niels Provos. Arpd. Retrieved May 16th 2003 from the World

[8] Wide http://www.citi.umich.edu/u/provos/honeyd/

[9] Net¯lter Core Team. ptables. Retrieved May 16th 2003 from the World Wide

[10] Web: http://www.net¯lter.org

[11] Martin Roesch. Snort. Retrieved May 17th 2003 from the World Wide Web:

[12] http://www.snort.org

[13] Honeynet Research Alliance. Honeynet Project Snort con¯gura-tion ¯le. Retrieved May 17th 2003 from the World Wide Web:

[14] http://www.honeynet.org/papers/honeynet/tools/snort.conf.

[15] Lance Spitzner Honeypots: Tracking Hackers September 10, 2002 Addison-Wesley Professional

[16] James O'Toole / Pittsburgh Post-Gazette Specter revises early attack ad against possible rival Toomey April 4, 2009

[17] Niels Provos, Thorsten Holz Virtual Honeypots: From Botnet Tracking to Intrusion Detection. July 16, 2007

[18] TANUSHA A Firefox Extension for Detecting Stored Cross Site Scripting Attack MARCH 23, 2011

[19] Stuart McDonald SQL Injection: Modes of Attack, Defence, and Why it matters April 8, 2002

[20] Lance Spitzner Honeytokens: The Other Honeypot 17 July, 2003

[21] Sainath Patil Nageshri B Karhade Honeyweb: a web-based high interaction client honeypot -30 Mar'12

[22] L. Spitzner, (2002). Honeypots: Tracking Hackers. 1st edition. Addison-Wesley Professiona. ISBN-10: 0321108957.

[23] Lance Spitzner. Honeypots: Catching the insider threat. In ACSAC '03: Proceedings of the 19th Annual Computer Security Applications Conference, page 170, Washington, DC, USA,2003. IEEE Computer Society.

[24] Fabien Pouget and Marc Dacier. Honeypot-based forensics. In AusCERT Asia Pacific Information technology Security Conference 2004, Brisbane, Australia, May 2004.

[25] BRUCE PERENS ,PHP 5 Power Programming y, September 23,