Brute Force Attacks

Unlike hacks that focus on vulnerabilities in software, a Brute Force Attack aims at being the simplest kind of method to gain access to a site: it tries usernames and passwords, over and over again, until it gets in. Often deemed 'inelegant', they can be very successful when people use passwords like '123456' and usernames like 'admin.'

They are, in short, an attack on the weakest link in any website's security... you.

Due to the nature of these attacks, you may find your server's memory goes through the roof, causing performance problems. This is because the number of http requests (that is the number of times someone visits your site) is so high that servers run out of memory.

This sort of attack is not endemic to WordPress, it happens with every webapp out there, but WordPress is popular and thus a frequent target.

Throttling Multiple Login Attempts

One of the most common kinds of attacks targeting internet services is brute force login attacks. With this form of attack, a malicious party tries to guess WordPress usernames and passwords. The attacker needs only the URL of a user site to perform an attack. Software is readily available to perform these attacks using botnets, making increasingly complex passwords easier to find.

The best protection against this kind of attack is to set and recommend and/or enforce strong passwords for WordPress users.

It is also recommended for hosts to throttle login attempts at the network and server level when possible. It's helpful to throttle both maximum logins per site over time, and maximum attempts per IP over time across server or infrastructure to mitigate bot password brute-force attacks. This can be done at the plugin level as well, but not without incurring the additional resource utilization caused during these attacks.

Protect Yourself

A common attack point on WordPress is to hammer the `wp-login.php` file over and over until they get in or the server dies. You can do some things to protect yourself.

Don't use the 'admin' username

The majority of attacks assume people are using the username 'admin' due to the fact that early versions of WordPress defaulted to this. If you are still using this username, make a new account, transfer all the posts to that account, and change 'admin' to a subscriber (or delete it entirely).

You can also use the plugin Change Username to change your username.

Good Passwords

The goal with your password is to make it hard for other people to guess and hard for a brute force attack to succeed. Many automatic password generators are available that can be used to create secure passwords.

WordPress also features a password strength meter which is shown when changing your password in WordPress. Use this when changing your password to ensure its strength is adequate.

You can use the Force Strong Password plugin to force users to set strong passwords.

Things to avoid when choosing a password:

- Any permutation of your own real name, username, company name, or name of your website.
- A word from a dictionary, in any language.
- A short password.
- Any numeric-only or alphabetic-only password (a mixture of both is best).

A strong password is necessary not just to protect your blog content. A hacker who gains access to your administrator account is able to install malicious scripts that can potentially compromise your entire server.

To further increase the strength of your password, you can enable Two Step Authentication to further protect your blog.

Plugins

There are many plugins available to limit the number of login attempts made on your site. Alternatively, there are also many plugins you can use to block people from accessing wp-admin altogether.

Protect Your Server

If you decide to lock down wp-login.php or wp-admin, you may find you get a 404 or 401 error when accessing these pages

If you decide to lock down wp-login.php or wp-admin, you may find you get a 404 or 401 error when accessing those pages. To avoid that, you will need to add the following to your .htaccess file.

```
ErrorDocument 401 default
```

You can have the 401 point to 401.html, but the point is to aim it at not WordPress.

For Nginx you can use the `error_page` directive but must supply an absolute url.

```
error_page  401  https://example.com/forbidden.html;
```

On IIS web servers you can use the `httpErrors` element in your web.config, set `errorMode="custom"`:

```
<httpErrors errorMode="Custom">
    <error statusCode="401"
    subStatusCode="2"
    prefixLanguageFilePath=""
    path="401.htm"
    responseMode="File" />
</httpErrors>
```

Password Protect wp-login.php

Password protecting your wp-login.php file (and wp-admin folder) can add an extra layer to your server. Because password protecting wp-admin can break any plugin that uses ajax on the front end, it's usually sufficient to just protect wp-login.php.

To do this, you will need to create a .htpasswd file. Many hosts have tools to do this for you, but if you have to do it manually, you can use this htpasswd generator. Much like your .htaccess file (which is a file that is only an extension), .htpasswd will also have no prefix.

You can either put this file outside of your public web folder (i.e. not in /public_html/ or /domain.com/, depending on your host), or you can put it in the same folder, but you'll want to do some extra security work in your .htaccess file if you do.

Speaking of which, once you've uploaded the .htpasswd file, you need to tell .htaccess where it's at. Assuming you've put .htpasswd in your user's home directory and your htpasswd username is mysecretuser, then you put this in your .htaccess:

```
# Stop Apache from serving .ht* files
<Files ~ "^\\.ht">
    Order allow,deny
    Deny from all
</Files>

# Protect wp-login.php
<Files wp-login.php>
    AuthUserFile ~/.htpasswd
    AuthName "Private access"
    AuthType Basic
    require user mysecretuser
</Files>
```

The actual location of AuthUserFile depends on your server, and the 'require user' will change based on what username you pick.

If you are using Nginx you can password protect your wp-login.php file using the HttpAuthBasicModule. This block should be inside your server block.

```
location /wp-login.php {
    auth_basic "Administrator Login";
    auth_basic_user_file .htpasswd;
}
```

The filename path is relative to directory of nginx configuration file nginx.conf

The file should be in the following format:

```
user:pass
user2:pass2
user3:pass3
```

Unfortunately there is no easy way of configuring a password protected wp-login.php on Windows Server IIS. If you use a .htaccess processor like Helicon Ape, you can use the .htaccess example mentioned above. Otherwise you'd have to ask your hosting provider to set up Basic Authentication.

All passwords must be encoded by function `crypt(3)`. You can use an online htpasswd generator to encrypt your password.

Throttle Multiple Login Attempts

One of the most common kinds of attacks targeting internet services is brute force login attacks. With this form of attack, a malicious party tries to guess WordPress usernames and passwords. The attacker needs only the URL of a user site to perform

malicious party tries to guess WordPress usernames and passwords. The attacker needs only the URL of a user site to perform an attack. Software is readily available to perform these attacks using botnets, making increasingly complex passwords easier to find.

The best protection against this kind of attack is to set and recommend and/or enforce strong passwords for WordPress users.

It is also recommended for hosts to throttle login attempts at the network and server level when possible. It's helpful to throttle both maximum logins per site over time, and maximum attempts per IP over time across server or infrastructure to mitigate bot password brute-force attacks. This can be done at the plugin level as well, but not without incurring the additional resource utilization caused during these attacks.

Limit Access to wp-login.php by IP

If you are the only person who needs to login to your Admin area and you have a fixed IP address, you can deny wp-login.php (and thus the wp-admin/ folder) access to everyone but yourself via an .htaccess or web.config file. This is often referred to as an IP whitelist.

Note: Beware your ISP or computer may be changing your IP address frequently, this is called dynamic IP addressing, rather than fixed IP addressing. This could be used for a variety of reasons, such as saving money. If you suspect this to be the case, find out out how change your computer's settings, or contact your ISP to obtain a fixed address, in order to use this procedure.

In all examples you have to replace 203.0.113.15 with your IP address. Your Internet Provider can help you to establish your IP address. Or you can use an online service such as What Is My IP.

Examples for multiple IP addresses are also provided. They're ideal if you use more than one internet provider, if you have a small pool of IP addresses or when you have a couple of people that are allowed access to your site's Dashboard.

Create a file in a plain text editor called .htaccess and add:

```
# Block access to wp-login.php.
<Files wp-login.php>
    order deny,allow
    allow from 203.0.113.15
    deny from all
</Files>
```

You can add more than one allowed IP address using:

```
# Block access to wp-login.php.
<Files wp-login.php>
    order deny,allow
    allow from 203.0.113.15
    allow from 203.0.113.16
    allow from 203.0.113.17
    deny from all
</Files>
```

Are you using Apache 2.4 and Apache module mod_authz_host? Then you have to use a slightly different syntax:

```
# Block access to wp-login.php.
<Files wp-login.php>
    Require ip 203.0.113.15
</Files>
```

If you want to add more than one IP address, you can use:

```
# Block access to wp-login.php.
<Files wp-login.php>
    Require ip 203.0.113.15 203.0.113.16 203.0.113.17
    # or for the entire network:
    # Require ip 203.0.113.0/255.255.255.0
</Files>
```

For Nginx you can add a location block inside your server block that works the same as the Apache example above.

```
error_page  403  https://example.com/forbidden.html;
location /wp-login.php {
    allow   203.0.113.15
    # or for the entire network:
    # allow   203.0.113.0/24;
    deny    all;
}
```

Note that the order of the deny/allow is of the utmost importance. You might be tempted to think that you can switch the access directives order and everything will work. In fact it doesn't. Switching the order in the above example has the result of denying access to all addresses.

Again, on IIS web servers you can use a web.config file to limit IP addresses that have access. It's best to add this in an additional `<location` directive.

```
<location path="wp-admin">
    <system.webServer>
        <security>
            <ipSecurity allowUnlisted="false"> <!-- this rule denies all IP addresses, except
the ones mentioned below -->
                <!-- 203.0.113.x is a special test range for IP addresses -->
                <!-- replace them with your own -->
                <add ipAddress="203.0.113.15" allowed="true" />
                <add ipAddress="203.0.113.16" allowed="true" />
            </ipSecurity>
        </security>
    </system.webServer>
</location>
```

Deny Access to No Referrer Requests

Extended from Combatting Comment Spam, you can use this to prevent anyone who isn't submitting the login form from accessing it:

```
# Stop spam attack logins and comments
<IfModule mod_rewrite.c>
    RewriteEngine On
    RewriteCond %{REQUEST_METHOD} POST
    RewriteCond %{REQUEST_URI} .(wp-comments-post|wp-login)\.php*
    RewriteCond %{HTTP_REFERER} !.*example.com.* [OR]
    RewriteCond %{HTTP_USER_AGENT} ^$
    RewriteRule (.*) https://%{REMOTE_ADDR}/$1 [R=301,L]
</ifModule>
```

Nginx – Deny Access to No Referrer Requests

```
location ~* (wp-comments-posts|wp-login)\\.php$ {
    if ($http_referer !~ ^(https://example.com) ) {
        return 405;
    }
}
```

Windows Server IIS – Deny access to no referrer requests:

```
<rule name="block_comments_without_referer" patternSyntax="ECMAScript" stopProcessing="true">
<match url="(.*)" ignoreCase="true" />
    <conditions logicalGrouping="MatchAll">
        <add input="{URL}" pattern="^/(wp-comments-post|wp-login)\.php" negate="false"/>
        <add input="{HTTP_REFERER}" pattern=".*example\.com.*" negate="true" />
        <add input="{HTTP_METHOD}" pattern="POST" />
    </conditions>
    <action type="CustomResponse" statusCode="403" statusReason="Forbidden: Access is denied."
 statusDescription="No comments without referrer!" />
</rule>
```

Change example.com to your domain. If you're using Multisite with mapped domains, you'll want to change example.com to `(example.com|example.net|example.org)` and so on. If you are using Jetpack comments, don't forget to add jetpack.wordpress.com as referrer: `(example.com|jetpack\.wordpress\com)`

ModSecurity

If you use ModSecurity, you can follow the advice from Frameloss – Stopping brute force logins against WordPress. This requires root level access to your server, and may need the assistance of your webhost.

If you're using ModSecurity 2.7.3, you can add the rules into your .htaccess file instead.

Fail2Ban

Fail2ban is a Python daemon that runs in the background. It checks the logfiles that are generated by Apache (or SSH for example), and on certain events can add a firewall rule. It uses a so called filter with a regular expression. If that regular expression happens for example 5 times in 5 minutes, it can block that IP address for 60 minutes (or any other set of numbers).

Installing and setting up Fail2ban requires root access.

Blocklists

It appears that most brute force attacks are from hosts from Russia, Kazachstan and Ukraine. You can choose to block ip-addresses that originate from these countries. There are blocklists available on the internet that you can download. With some shell-scripting, you can then load blockrules with iptables.

You have to be aware that you are blocking legitimate users as well as attackers. Make sure you can support and explain that decision to your customers.

Besides blocklists per country, there are lists with ip-addresses of well-known spammers. You can also use these to block them with iptables. It's good to update these lists regularly.

Setting up of blocklists and iptables requires root access.

Setting up of blocklists and iptables requires root access.

Cloud/Proxy Services

Services like CloudFlare and Sucuri CloudProxy can also help mitigate these attacks by blocking the IPs before they reach your server.

See Also

- Sucuri: Protecting Against WordPress Brute Force Attacks
- How to: Protect WordPress from brute-force XML-RPC attacks
- Liquid Web: ModSecurity Rules To Alleviate Brute Force Attacks
- Swiss Army Knife for WordPress (SAK4WP) – Free Open Source Tool that can help you protect your wp-login.php and /wp-admin/ but not /wp-admin/admin-ajax.php with one click and much more

Changelog

- 2022-10-25: Original content from Brute Force Attacks.

First published

March 28, 2023

Last updated

January 16, 2024

Previous
HTTPS

Next
Hardening WordPress