Wordfence
Critical Authentication Bypass Vulnerability Patched in SiteGroun
Security Plugin

Chloe Chamberland April 7, 2022

Critical Authentication Bypass Vulnerability Patched in SiteGround Security Plugin

On March 10, 2022 the Wordfence Threat Intelligence team initiated the responsible disclosure process for a vulnerability we discovered in “SiteGround Security”, a WordPress plugin that is installed on over 400,000 sites. This f makes it possible for attackers to gain administrative user access on vulnerable sites when two-factor authenticatior (2FA) is enabled but not yet configured for an administrator.

Wordfence Premium, Wordfence Care, and Wordfence Response received a set of firewall rules on March 10, 2022 tc provide protection against any attackers trying to exploit this vulnerability. Wordfence Free users will receive this sam protection 30 days later on April 9, 2022

After sending the full disclosure details to the SiteGround security team on March 10, 2022 a patch was released the next day on March 11, 2022. While the plugin was partially patched immediately, it wasn’t optimally patched until Apr 2022.

Sites hosted on the SiteGround platform have automatically been updated to the patched version while those hosted elsewhere will require a manual update, if auto-updates are not enabled for the plugin. We strongly recommend ensu that your site has been updated to the latest patched version of “SiteGround Security”, which is version 1.2.6 at the tir of this publication.

Description: Authentication Bypass via 2-Factor Authentication Setup
Affected Plugin: SiteGround Security
Plugin Slug: sg-security
Plugin Developer: SiteGround
Affected Versions: <= 1.2.5
CVE ID: CVE-2022-0992
CVSS Score: 9.8 (Critical)

CVSS Vector: CVSS:3.1/AV:N/AC:L/PR:N/UI:N/S:U/C:H/I:H/A:H
Researcher/s: Chloe Chamberland

login security including 2FA, general WordPress hardening, activity monitoring, and more. It's also worth noting that it comes pre-installed on all SiteGround hosted WordPress sites. Unfortunately, the 2FA functionality of the plugin was insecurely implemented making it possible for unauthenticated attackers to gain access to privileged accounts.

When two-factor authentication is enabled, it requires all administrative and editor users to set-up two factor authentication. This requirement is triggered when the site's administrative and editor users log into the site for the fi time after 2FA has been enabled at which time they are prompted to configure 2FA for their account. This means tha there will be a period of time between 2FA being enabled on a site and each user configuring it for the account.

During this interim period, attackers could hijack the 2FA set-up process. The plugin had a flaw that made it so that attackers could completely bypass the first step of authentication, which requires a username and password, and access the 2FA set-up page for users that had not configured 2FA yet.

It was as simple as supplying the user ID they would like to compromise via the `sg-user-id` parameter, along with a other parameters to indicate that they would like to trigger the initial 2FA configuration process.

The following `validate_2fa_login()` function shows the process by which a user-supplied ID is validated. If the res from the `check_authentication_code()` function and the `sg_security_2fa_configured` user meta retuned false, which indicated that 2FA hasn't yet been configured for that user, then the plugin would load the 2fa-initial-setup-form.php template which displays the QR code and 2FA secret needed to configure the authenticator app for the use supplied ID.

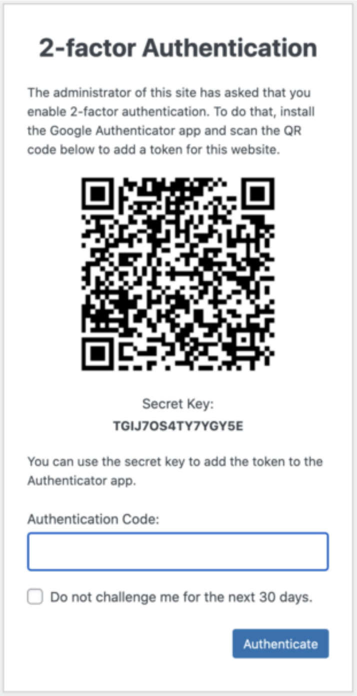
```
</pre>
<pre>public function validate_2fa_login( $user ) {
   // Bail if there is no valid user authentication.
   if ( ! isset( $_POST['sg-user-id'] ) ) { // phpcs:ignore
      return;
   }

   $result = $this->check_authentication_code( wp_unslash( $_POST['sgc2facode'] ), wp_unslash( $_POST['sg-user-ic

   // Check the result of the authtication.
   if ( false === $result ) {
      if ( 0 == get_user_meta( $_POST['sg-user-id'], 'sg_security_2fa_configured', true ) ) { // phpcs:ignore
         // Arguments for initial 2fa setup.
         $args = array(
            'template' => '2fa-initial-setup-form.php',
            'qr'       => get_user_meta( $_POST['sg-user-id'], 'sg_security_2fa_qr', true ), // phpcs:ignore
            'secret'   => get_user_meta( $_POST['sg-user-id'], 'sg_security_2fa_secret', true ), // phpcs:ignore
            'error'    => esc_html__( 'Invalid verification code!', 'sg-security' ),
            'action'   => esc_url( add_query_arg( 'action', 'sgs2fa', wp_login_url() ) ),
         );
      } else {
         // Arguments for 2fa login.
         $args = array(
            'template' => '2fa-login.php',
            'error'    => esc_html__( 'Invalid verification code!', 'sg-security' ),
            'action'   => esc_url( add_query_arg( 'action', 'sgs2fa', wp_login_url() ) ),
         );
      }

      $this->load_form( wp_unslash( $_POST['sg-user-id'] ), $args ); // phpcs:ignore
   }

   // Set the auth cookie.
   wp_set_auth_cookie( wp_unslash( $_POST['sg-user-id'] ), intval( wp_unslash( $_POST['rememberme'] ) ) ); // phr
<pre>
```



2-factor Authentication

The administrator of this site has asked that you enable 2-factor authentication. To do that, install the Google Authenticator app and scan the QR code below to add a token for this website.

Secret Key:
TGIJ7OS4TY7YGY5E

You can use the secret key to add the token to the Authenticator app.

Authentication Code:

Do not challenge me for the next 30 days.

Authenticate

The authentication QR code and secret key displayed that would be displayed to potentially unauthorized users.

The returned QR code and secret key are the only things needed to connect the user account with an authentication mechanism, such as Google Authenticator. Attackers were able to use this to connect their authentication app with tl account and successfully use a code to pass the "second factor of authentication." This function would then set the u authentication cookies via the `wp_set_auth_cookie()` function using the user supplied ID from the `sg-user-id` parameter which effectively logs the attacker in as that user. Due to the default configuration of the plugin, this accou would most likely be a privileged user like an administrator or editor. It's also worth noting that the function returns th back-up codes which could be used via the weakness outlined in the next section.

To sum it up, there was no validation on the `validate_2fa_login()` function that the identity a user was claiming w in fact legitimate. As such attackers could bypass the first authentication mechanism, a username/password pair, w is meant to prove identity and successfully log in, due to a weakness in the second authentication mechanism, the 2I process. When successful, an attacker could completely infect a site by exploiting this vulnerability.

Description: Authorization Weakness to Authentication Bypass via 2-Factor Authentication Back-up Codes
Affected Plugin: SiteGround Security
Plugin Slug: sg-security
Plugin Developer: SiteGround
Affected Versions: <= 1.2.4
CVE ID: CVE-2022-0993
CVSS Score: 8.1 (High)
CVSS Vector: CVSS:3.1/AV:N/AC:H/PR:N/UI:N/S:U/C:H/I:H/A:H
Researcher/s: Chloe Chamberland
Fully Patched Version: 1.2.6

In addition to the above outlined vulnerability, the method in which 2FA back-up code authentication was handled ma it possible for attackers to log in if they were able to brute force a back-up code for a user or compromise it via other means such as SQL Injection.

Diving deeper, the plugin registered the `validate_2fabc_login()` function which validated the supplied backup code through the `validate_backup_login()` function using the user supplied user ID from the `sg-user-id` parameter alc with the back-up code supplied via the `sgc2fabackupcode` parameter. If the back-up code was found in the array of stored back-up codes for that user, then the function would use the `wp_set_auth_cookie()` function to set the authentication cookies for the supplied user ID. If that user ID belonged to an administrator, the attacker would effectively be logged in as an administrator.

```
</pre>
<pre>public function validate_2fabc_login() {
```

```
   if ( false === $result ) {
      $this->load_form(
         wp_unslash( $_POST['sg-user-id'] ), // phpcs:ignore
         array(
            'template' => '2fa-login-backup-code.php',
            'action'   => esc_url( add_query_arg( 'action', 'sgs2fabc', wp_login_url() ) ),
            'error'    => esc_html__( 'Invalid backup code!', 'sg-security' ),
         )
      );
   }

   // Set the auth cookie.
   wp_set_auth_cookie( wp_unslash( $_POST['sg-user-id'] ), intval( wp_unslash( $_POST['rememberme'] ) ) ); // php
```

Similarly to the previous vulnerability, the issue here is that there was no true identity validation for the authentication which indicates an authorization weakness. The function performed no checks to verify that a user had previously authenticated prior to entering the 2FA back-up code, and as such they did not need to legitimately log in prior to beir logged in while using a back-up code. This meant that there were no checks to validate that a user was authorized to use a back-up code to perform the second factor of authentication that would log them in.

Though the risk in this case is lower, the backup codes were 8 digits long and entirely numeric, so an attacker could potentially brute force one of the 8 back-up codes and automatically be logged in without knowing a username and password combination for an administrative user.

While this might not be practical to attempt on most servers, a patient adversary attacking a well-provisioned server capable of processing a large number of requests at once would have a high chance of eventually gaining access un the brute force attempts were stopped by another mechanism, such as the Wordfence plugin's built-in brute force protection or rate limiting rules.

Further, this vulnerability could be used in conjunction with another vulnerability, such as SQL injection, where an attacker would be able to compromise the 2FA back-up codes that are stored in the database and then subsequently use them to log in without needing to crack the password of an administrative user which would likely be significantl stronger. In both cases, the impact would be significant as an attacker could gain administrative access to the compromised WordPress site which could be used for complete site infection.

An Important Security Reminder: Audit Your WordPress Site's User Accounts

This vulnerability serves as an important reminder to audit your WordPress site's user accounts. This means identifyi any old and unused user accounts that have been inactive for an extended period of time and/or are likely to never be used again and removing them or completely stripping the user's capabilities. This vulnerability could easily be exploi on sites where the site owner enabled 2FA, which is required for all administrative and editor users, and had old inact administrative/editor user accounts on the site that an attacker could target. Considering accounts that are no longe active are unlikely to log in after the 2FA setting has been enabled, the 2FA for those accounts would not be configure leaving the site ripe for exploitation by any attackers exploiting the vulnerability.

A situation involving a similar security issue involving insecure 2FA was reported by the CISA in conjunction with the a few weeks ago, around the same time we discovered this vulnerability. In the Cybersecurity Advisory (CSA) by the CISA, it was disclosed that a threat actor was able to successfully brute force a dormant user's account credentials, a due to a default 2FA setting that would allow dormant users to re-enroll a new device for 2FA during the next active lo in, the threat actor was able to connect the 2FA secret to their own account and retrieve the code needed to pass the second factor of authentication. Once the threat actor gained initial access to the system they were able to escalate their privileges by exploiting the "PrintNightmare" vulnerability, which you can read more about here, and steal sensiti information from across the organization's network. This goes to show that attackers are definitely looking for flaws the one disclosed today to exploit and any site can be a target. As such, it's important to actively maintain and valida the security of your site through regularly performed professional or self-conducted security audits and penetration tests, which is a service Wordfence provides. Security is an active and continuous process.

Timeline

March 10, 2022 – Conclusion of the plugin analysis that led to the discovery of two Authentication Bypass

March 11, 2022 – The CTO of SiteGround responds indicating that a patch has been released. We review the patch a inform them that it is insufficient. They release an additional patch.
March 11, 2022 – A patched version of the plugin is released as version 1.2.3. We suggest further security enhancements to the functionality.
March 16, 2022 – An update is made that reduces the security of the 2FA functionality, we follow-up again to sugges better security enhancements to the functionality. The CTO assures us that they are working on it.
April 6, 2022 – A fully and optimally patched version of the plugin is released as version 1.2.6.
April 9, 2022 – Wordfence Free users receive the firewall rules.

Conclusion

In today's post, we detailed a flaw in the "SiteGround Security" plugin that made it possible for unauthenticated attack to gain access to administrative user accounts in instances where 2-Factor Authentication was enabled, though not y fully set up, and in cases where an attacker could successfully brute force a back-up code. This could easily be used an attacker to completely compromise a site. This flaw has been fully patched in version 1.2.6.

We strongly recommend ensuring that your site has been updated to the latest patched version of "SiteGround Secur which is version 1.2.6 at the time of this publication.

Wordfence Premium, Wordfence Care, and Wordfence Response received a set of firewall rules on March 10, 2022 tc provide protection against attempts by attackers to exploit this vulnerability. Wordfence Free users will receive this sa protection 30 days later on April 9, 2022

If you believe your site has been compromised as a result of this vulnerability or any other vulnerability, we offer Incid 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 Wordfence Care and Wordfence Response include hands-on security support that provide you with ongoing assistance from our incident response team, shoulc you need it.

Special thanks to the team at SiteGround, for responding swiftly and working quickly to get a patch out to protect their customers and working to further secure the 2FA component.

Did you enjoy this post? Share it!

Comments

No Comments

Products

Wordfence Free
Wordfence Premium
Wordfence Care
Wordfence Response
Wordfence Central

Support

Documentation
Learning Center
Free Support
Premium Support

News

Blog
In The News
Vulnerability Advisories

About

About Wordfence
Careers
Contact
Security
CVE Request Form

Stay Updated

Sign up for news and updates from our panel of experienced security professionals.

you@example.com

☐ By checking this box I agree to the terms of service and privacy policy.*

SIGN UP