Ram Gall

August 20, 2020

High-Severity Vulnerability Patched in Advanced Access Manager

On August 13, 2020, the Wordfence Threat Intelligence team finished investigating two vulnerabilities in Advanced Access Manager, a WordPress plugin with over 100,000 installations, including a high-severity Authorization Bypass vulnerability that could lead to privilege escalation and site takeover.

We reached out to the plugin's author the next day, on August 14, 2020, and received a response within a few hours. After providing the full vulnerability disclosure, we received a response on August 15, 2020, that a patch had been released in version 6.6.2.

Wordfence Premium users received a firewall rule protecting against the Authorization Bypass vulnerability on August 14, 2020. Sites still running the free version of Wordfence will receive this rule 30 days later, on September 13, 2020.

Description: Authenticated Authorization Bypass and Privilege Escalation
Affected Plugin: Advanced Access Manager
Plugin Slug: advanced-access-manager
Affected Versions: < 6.6.2
CVE ID: CVE-2020-35935
CVSS Score: 7.5(High)
CVSS Vector: CVSS:3.0/AV:N/AC:H/PR:L/UI:N/S:U/C:H/I:H/A:H
Fully Patched Version: 6.6.2

Advanced Access Manager allows fine-grained access control, and has the capability to assign multiple roles to a single user. If the "Multiple Roles Support" setting is enabled, the plugin is vulnerable to authenticated authorization bypass and, in some cases, privilege escalation.

A low-privileged user could assign themselves or switch to any role with an equal or lesser user level, or any role that did not have an assigned user level. This could be done by sending a `POST` request to `wp-admin/profile.php` with typical profile update parameters and appending a `aam_user_roles[]` parameter set to the role they would like to use.

The reason this worked is that the `AAM_Backend_Manager::profileUpdate` method that actually assigns these roles is triggered by the `profile_update` and `user_register` actions, and failed to use a standard capability check.

```
add_action('profile_update', array($this, 'profileUpdate'), 10, 2);
add_action('user_register', array($this, 'profileUpdate'), 10, 2);
```

```
public function profileUpdate($id)
{
    $user = get_user_by('ID', $id);

    //save selected user roles
    if (AAM::api()->getConfig('core.settings.multiSubject', false)) {
        $roles = filter_input(
            INPUT_POST,
            'aam_user_roles',
            FILTER_DEFAULT,
            FILTER_REQUIRE_ARRAY
        );

        // let's make sure that the list of roles is array

        $roles = (is_array($roles) ? $roles : array());
        $okroles=array_keys(get_editable_roles());

        // prepare the final list of roles that needs to be set
        $newRoles = array_intersect($roles, array_keys(get_editable_roles()));

        if (!empty($newRoles)) {
            //remove all current roles and then set new
            $user->set_role('');

            foreach ($newRoles as $role) {
                $user->add_role($role);
            }
        }
    }
}
```

This meant that, if the 'Multiple Roles Support' setting was enabled, any user would trigger this method when updating their profile. The `profileUpdate` function would then check to see if any roles were present in the `aam_user_roles[]` `POST` parameter. If roles were present, it then used the WordPress `get_editable_roles` function to determine whether the user was allowed to add a given role, and if so, granted the user that role without performing any other form of capability check.

By default, `get_editable_roles` returns all registered roles. However, the Advanced Access Manager plugin added a filter to limit these roles in the `AAM_Service_UserLevelFilter::filterRoles` method. This method looped through each registered role and determined the role's user level using the `AAM_Core_API::maxLevel` method.

```
public function filterRoles($roles)
{
    static $levels = array(); // to speed-up the execution

    foreach ($roles as $id => $role) {
        if (!empty($role['capabilities']) && is_array($role['capabilities'])) {
            if (!isset($levels[$id])) {
                $levels[$id] = AAM_Core_API::maxLevel($role['capabilities']);
            }

            if (!$this->isUserLevelAllowed(true, $levels[$id])) {
                unset($roles[$id]);
            }
        }
    }

    return $roles;
}
```

`AAM_Service_UserLevelFilter::filterRoles` then removed any roles with a higher user level than the current user from the list of roles the current user was allowed to choose. By default this worked reasonably well all built-in roles

This meant that if a role did not have a user-level attribute, or had a user-level attribute equal to or lesser than the logged-in user, the logged-in user could assign themselves to that role.

This was a problem in 3 possible scenarios:

- **Plugins with custom roles.** There are several thousand plugins that add custom roles in the WordPress plugin repository, and most of these plugins do not assign a user-level attribute to these roles. For a few real-world examples, a backup plugin could add a role that is allowed to restore arbitrary files, including malicious code or database modifications, or an educational plugin might add an instructor role with the ability to insert unfiltered html and embed malicious JavaScript into the site.
- **Roles without an assigned user level**. If a role was created from scratch in Advanced Access Manager, but not assigned a user level, any user with subscriber-level access could switch to that role.
- **Cloned user roles.** If a role was cloned from an existing role (for instance, a contributor or author) and assigned additional capabilities, any user in the original role could switch to or assign themselves to the new role.

In any one of these scenarios, a low-privileged attacker could potentially switch to a role that allowed them to either directly take over a site or could be used as part of an exploit chain, depending on which roles were configured.

Description: Authenticated Information Disclosure
Affected Plugin: Advanced Access Manager
Plugin Slug: advanced-access-manager
Affected Versions: < 6.6.2
CVE ID: CVE-2020-35934
CVSS Score: 4.3(Medium)
CVSS Vector: CVSS:3.0/AV:N/AC:L/PR:L/UI:N/S:U/C:L/I:N/A:N
Fully Patched Version: 6.6.2

Advanced Access Manager also allows users to login via the WordPress REST API. Unfortunately the plugin's `aam/v1/authenticate` and `aam/v2/authenticate` REST endpoints were set to respond to a successful login with a json-encoded copy of all metadata about the user, potentially exposing users' information to an attacker or low-privileged user. This included items like the user's hashed password and their capabilities and roles, as well as any custom metadata that might have been added by other plugins. This might include sensitive configuration information, which an attacker could potentially use as part of an exploit chain. For example, an attacker able to assign themselves a custom role using the previous vulnerability could view which capabilities were assigned to them, allowing them to plan the next phase of their attack.

What are Roles and Capabilities?

All WordPress sites need an administrator, a user who has complete control over the site in order to make changes and perform maintenance. Likewise, an eCommerce site would need to allow customers to log in to keep track of their orders, and a news site would likely need to allow its journalists the ability to author posts, and might need an editor. In these cases, "administrator", "editor", "author", and "customer" are **roles**.

Each of these roles comes with a certain set of **capabilities**. For example, an administrator would have the `manage_options` capability that allows them to make changes to a site's options, but it would be disastrous to allow a customer, or even an author, the same capabilities. Likewise, an author would need the capability to `edit_posts`, but not `edit_others_posts`, and a customer or subscriber should not have any of these capabilities.

In many cases, a site owner might need more fine-grained control over which users can perform certain actions, so they might use a plugin like Advanced Access Manager to create custom roles for their users. They could then assign the capabilities they want those users to have, such as allowing a site designer the ability to `edit_theme_options` without changing other site options.

Additionally, many plugins add specific roles and custom capabilities. For example, while "customer" is not a built-in WordPress role, eCommerce plugins will define a specific "customer" role that has custom capabilities related to viewing their order status and updating their address information, while prohibiting them from making any other changes to the site.

The ability to customize roles and capabilities using plugins is part of the power of using WordPress for a variety of applications, including eCommerce, learning management systems, membership sites, and many others. However, this expanded functionality demands a greater attention to access control and capabilities from site owners.

Timeline

August 13, 2020 – Wordfence Threat Intelligence finishes analyzing vulnerabilities.
August 14, 2020 – We release a Firewall rule to Wordfence Premium users to protect against privilege escalation vulnerability and provide disclosure to the plugin's author.
August 15, 2020 – A full patch is released.
September 13, 2020 – Firewall rule becomes available to sites using the free version of Wordfence.

Conclusion

In today's post, we detailed two vulnerabilities in the Advanced Access Manager plugin, including a high-severity vulnerability that could allow lower-level users to escalate their privileges. We strongly recommend updating to the latest version of the Advanced Access Manager plugin, currently version 6.6.2, as soon as possible.

Wordfence Premium users have been protected against this vulnerability since August 14, 2020. Sites still using the free version of Wordfence will receive the firewall rule 30 days later, on September 13, 2020.

If you know a friend or colleague who is using this plugin on their site, please forward this advisory to them to help keep their sites protected.

Special thanks to the plugin's author, Vasyl Martyniuk, for an excellent and rapid response to our disclosure
Did you enjoy this post? Share it!

Comments

2 Comments

Yvonne Finn *
August 20, 2020
12:27 pm

Wordfence free or premium is the best security a WordPress website can rely on.
The speed and effectiveness with which they handle security threats is incomparable.
Thank you Wordfence, you deserve all the accolades you receive!
Yvonne Finn

Ram Gall *
August 21, 2020
7:11 am

Hi Yvonne,

Thank you so much for your kind words!

Breaking WordPress Security Research in your inbox as it happens.

you@example.com

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

SIGN UP

Our business hours are 9am-8pm ET, 6am-5pm PT and 2pm-1am UTC/GMT excluding weekends and holidays.
Response customers receive 24-hour support, 365 days a year, with a 1-hour response time.

Terms of Service Privacy Policy

CCPA Privacy Notice

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