WordPress Security Controls

List of hooks used by WordPress Security Controls modules

The underlying logic for the WordPress Security Controls feature is open source and available in the VIP Security Boost GitHub repository. We've compiled a list of all the WordPress hooks we use in our modules to help you identify and resolve any potential conflicts with your own custom code.

We recommend using the WordPress Security Controls feature in the VIP Dashboard to configure security settings for your WordPress application rather than relying on code-based approaches. The WordPress Security Controls feature will be available starting **Wednesday, August 20, 2025**, making it easy to manage security settings without custom code.

Inactive Users

You can review the specific logic for the Inactive Users module on GitHub.

Filters:

- `determine_current_user`
- `authenticate`
- `wp_is_application_passwords_available_for_user`
- `rest_authentication_error`
- `users_list_table_query_args`
- `vip_site_details_index_data`

Actions:

- `admin_init`
- `add_user_to_blog`
- `user_register`
- `vip_support_user_added`
- `admin_head-users.php`
- `admin_head-users-network.php`

User page view/columns:

- `wpmu_users_columns`
- `manage_users_columns`
- `manage_users_custom_column`
- `manage_users_sortable_columns`
- `manage_users-network_sortable_columns`

- `views_users`
- `views_users-network`

If you're using the Admin and Site Enhancements (ASE) plugin or the Admin Columns Pro plugin to modify the users list, certain WordPress Security Controls functionality may not appear as expected.

How to override

This security setting relies on around 20 different WordPress hooks. Because of this breadth, a reliable code-level override isn't practical. If you need different behavior, please contact Support and we can disable this specific setting for your environment.

Note that the "Inactive Users" module in the VIP Dashboard WordPress Security Controls page will not automatically sync with your custom logic or display a warning if you have custom code implemented for this module.

Enforce 2FA

You can review the specific logic for the Enforce 2FA module on GitHub.

Filters:

- `wpcom_vip_is_two_factor_forced`

Actions:

- `set_current_user`

If you use the `wpcom_vip_enable_two_factor` filter, there may be conflicts with the WordPress Security Controls functionality.

How to override

If you are using the `wpcom_vip_is_two_factor_forced` filter to enforce 2FA for:

- **All users:** No changes are required. Your existing code will continue to work.
- **Specific roles or capabilities**: Your existing code will continue to work. We suggest configuring enforcement through WordPress Security Controls or adopting our new filter:
  - **Current filter:** `wpcom_vip_is_two_factor_forced` supports roles and capabilities; will override settings enforced by WordPress Security Controls.
  - **New filter:** `wpcom_vip_wsc_forced_mfa_users_additional_capabilities` supports capabilities only; works in tandem with settings enforced by WordPress Security Controls.

- **Disable 2FA for all users:** No changes are required. Your existing code will continue to work. We strongly advise against disabling 2FA entirely; if you need more granular control, we recommend managing enforcement through the WordPress Security Controls feature.

The `wpcom_vip_is_two_factor_forced` filter will be deprecated in the future. We recommend configuring your enforcements through WordPress Security Controls in the VIP Dashboard.

```php
// Example usage: wpcom_vip_wsc_forced_mfa_users_additional_capabilities filter

// Enforce 2FA for settings enforced by WordPress Security Controls, plus everyone who can edit private pages and upload files.
add_filter( 'wpcom_vip_wsc_forced_mfa_users_additional_capabilities', function () {

    return [ 'edit_private_pages', 'upload_files' ];
} );
```

XML-RPC

You can review the specific logic for the XML-RPC module on GitHub.

Filters:

- `xmlrpc_enabled`
- `wp_headers`
- `xmlrpc_methods`
- `wp_xmlrpc_server_class`
- `authenticate`

Actions:

- `wp_head`

How to override

No code-based changes are required. Simply set the “XML-RPC” module to “Default protections” in the VIP Dashboard “WordPress Security Controls” page to disable our hooks.

WordPress Session Time

You can review the specific logic for the WordPress Session Time module on GitHub.

Filters:

- `auth_cookie_expiration`

How to override

Implement your custom logic using the `auth_cookie_expiration` filter and select the “Default” option in the “WordPress Session Time” module in the VIP Dashboard “WordPress Security Controls” page. The session time specified in your code will be enforced.

General Override

You can control when your custom code executes in relation to ours by setting your hook priority to run after muplugins_loaded:5 (e.g., with a priority of 6). The order of hooks in your custom code can affect how our hooks load, and your override may not work as intended.

```php
add_action(
    'muplugins_loaded', // Execute after all mu-plugins are loaded
    function() {
        add_filter(
            'auth_cookie_expiration', // Override the auth cookie expiration time
            function() {
                return 2 * DAY_IN_SECONDS;
            },
            100 // WordPress Security Controls implements it with priority 99, so use 100 to override it
        );
    },
    6 // Priority 6 makes sure this runs after WordPress Security Controls
);
```

Default settings for WordPress Security Controls

The following settings represent the default configuration for the new WordPress Security Controls feature. **You can modify these settings at any time in the VIP Dashboard with our new WordPress Security Controls feature**.

New environments

Starting on **Wednesday, August 20, 2025**, WordPress Security Controls will be enabled by default on all new production and non-production environments with the default settings outlined below.

- **Inactive Users**: `Administrator` users who have been inactive for 90 days or more will be flagged as inactive but will not be blocked.
- **Enforce 2FA**: Two-Factor Authentication (2FA) will be required to log in to the WP Admin Dashboard for all users with the `Administrator` and `Editor` roles.
- **XML-RPC**: XML-RPC requests will be allowed only via application passwords.
- **WP Session Time**: The default WordPress session timeout (14 days) will be applied.
- **Highlighted 2FA**: Users with `Administrator` and `Editor` roles who have 2FA disabled will be highlighted in a notice on the WP Admin Users page. This security setting is not configurable.

Existing environments

Non-Production: Starting on **Wednesday, August 20, 2025**, WordPress Security Controls will be enabled by default on all existing non-production environments with default settings. You can manually disable and enable again to ensure compatibility with custom code until enforcement begins starting October 1, 2025.

All environments: Starting **Wednesday, October 1, 2025**, **WordPress Security Controls will be enforced across *all* WordPress environments** with the default settings outlined below. Enablement will be rolled out in phases in the weeks following that date; exact timing will vary by environment. Any production or non-production environment that does not already have the feature enabled will be automatically updated. After this date, the feature cannot be disabled.

- **Inactive Users**: `Administrator` users who have been inactive for 90 days or more will be flagged as inactive but will not be blocked.
- **Enforce 2FA**: Two-Factor Authentication (2FA) will be required to log in to the WP Admin Dashboard for all users with the `Administrator` role.
- **XML-RPC**: XML-RPC requests will be allowed using a username and password, consistent with VIP's current default protections.
- **Limit Session Time**: The default WordPress session timeout (14 days) will apply.

- **Highlighted 2FA**: Users with `Administrator` and `Editor` roles who have 2FA disabled will be highlighted in a notice on the WP Admin Users page. This security setting is not configurable.