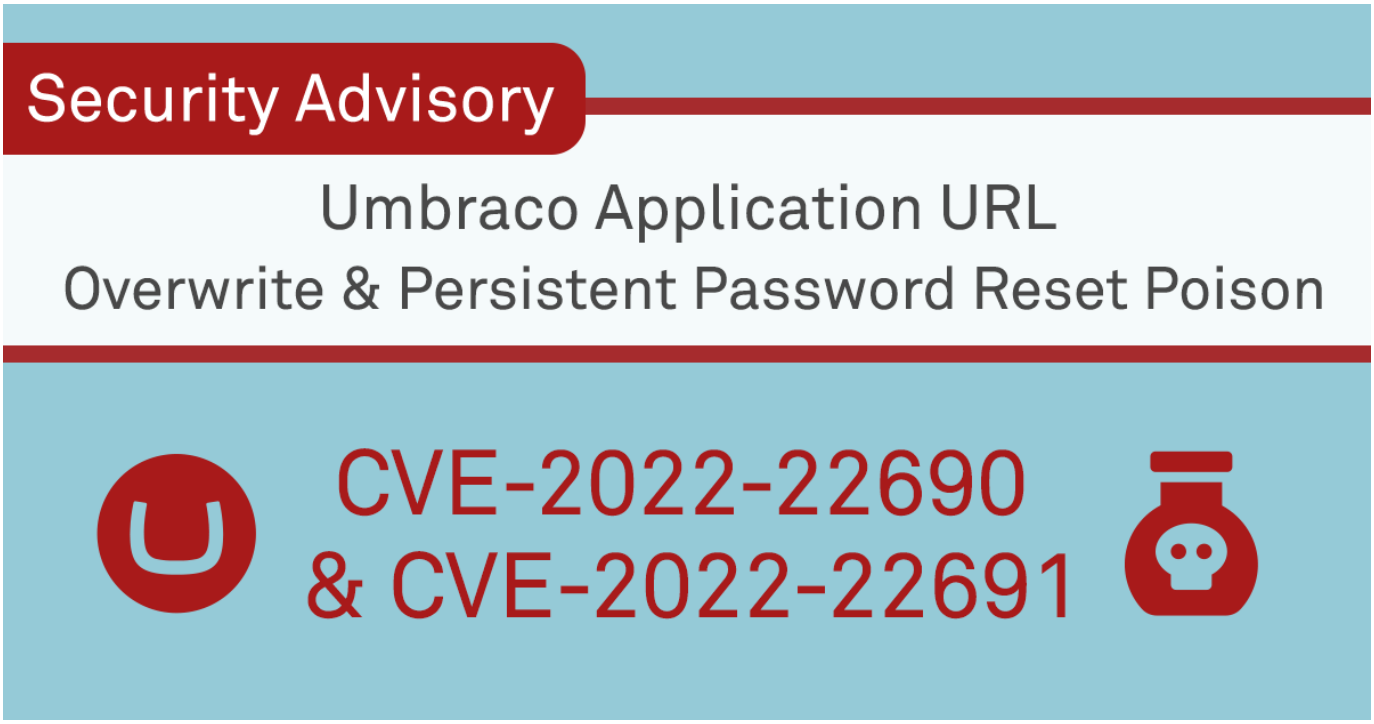

Umbraco ApplicationURL Overwrite & Persistent Password Reset Poison (CVE-2022-22690 & CVE-2022-22691)

Research / Security Alerts / Posted January 18, 2022

AppCheck Research identified multiple vulnerabilities within the Umbraco CMS that could be remotely exploited to persistently modify a sensitive configuration parameter used when generating URL's that reference the Umbraco application. The attacker could exploit this to poison password reset URL's and perform account take over attacks.

Overview

Within the Umbraco CMS, a configuration element named "*UmbracoApplicationUrl*" (or just "*ApplicationUrl*") is used whenever application code needs to build a URL pointing back to the site. For example, when a user resets their password and the application builds a password reset URL or when the administrator invites users to the site.

For Umbraco versions less than 9.2.0, if the Application URL is not specifically configured, the attacker can manipulate this value and store it persistently affecting all users for components where the "UmbracoApplicationUrl" is used (CVE-2022-22690).

For example, **the attacker is able to change the URL users receive when resetting their password so that it**

This website uses cookies to improve user experience. By using our website you consent to the storage of cookies from this website.
I agree

Analysis

(Note that the affected variable is referred to using slightly different names depending on where it is used. In code, “ApplicationUrl” is typically used and therefore we will use this name from here on).

The ApplicationUrl (The Base URL for the Umbraco install) is implemented in a number of areas where a URL needs to be built such as Password Reset, User Invite, automated health check processes and others.

In cases where the administrator has not specifically configured an Application URL, Umbraco falls back to the following code to enumerate and then cache the URL for future use **by all users**:

File: Umbraco.Web.Common\Middleware\UmbracoRequestMiddleware.cs

```
1.  public async Task InvokeAsync(HttpContext context, RequestDelegate next)
2.      //.. Truncated for brevity ..
3.
4.      Uri currentApplicationUrl = GetApplicationUrlFromCurrentRequest(context.Request);
5.      _hostingEnvironment.EnsureApplicationMainUrl(currentApplicationUrl);
6.      var pathAndQuery = context.Request.GetEncodedPathAndQuery();
```

The “**GetApplicationUrlFromCurrentRequest**” function call in the code above is responsible for enumerating the current URL based on the users’ inbound request. The code for this function is listed below, the key component from the attackers perspective is the “{request.Host}” variable, this is populated from *host* header of the inbound HTTP request.

The attacker can manipulate the URL returned by the *GetApplicationUrlFromCurrentRequest* function to point to his/her server by modifying the HTTP host header for any request to the Umbraco server.

```
1.  private Uri GetApplicationUrlFromCurrentRequest(HttpRequest request){
2.      // We only consider GET and POST.
3.      // Especially the DEBUG sent when debugging the application is annoying because it uses http, even when the
    https is available.
4.      if (request.Method == "GET" || request.Method == "POST")
5.      {
6.          return new Uri($"{request.Scheme}://{request.Host}{request.PathBase}", UriKind.Absolute);
7.      }
8.          return null;
9.  }
```

The attacker controlled URL value is then passed to the “***EnsureApplicationMainUrl***” function where it is cached via the “***_applicationUrls.TryAdd***” function and used as the ApplicationUrl from that point onward (until the server is restarted).

Note that the Umbraco installation will save a new ApplicationUrl in cases where it hasn’t previously seen the supplied value. This means that when the attacker manipulates the ApplicationUrl, it will not be changed back when a subsequent benign request is received since the legitimate host value will have already been previously observed.

For completeness the *EnsureApplicationMainUrl function code is listed below:*

```
1.  public void EnsureApplicationMainUrl(Uri currentApplicationUrl)
```

This website uses cookies to improve user experience. By using our website you consent to the storage of cookies from this website.

I agree

```
            if (currentApplicationUrl is null)
            {
                return;
            }

            if (_webRoutingSettings.CurrentValue.UmbracoApplicationUrl is not null)
            {
                return;
            }

            var change = !_applicationUrls.Contains(currentApplicationUrl);
            if (change)
            {
                if (_applicationUrls.TryAdd(currentApplicationUrl))
                {
                    ApplicationMainUrl = currentApplicationUrl;
                }
            }
        }
    }
```

Exploiting the Vulnerability

To exploit this flaw the attacker needs to deliver a request to the Umbraco CMS with an “Host” header value set to point to the attackers server. This is possible when the Microsoft IIS Server bindings are not specifically configured to lock the server down to a specific hostname.

Screenshot: Leaving this field blank will serve the site for any hostname and allows the flaw to be exploited.

A quick test for this configuration is to access your site via its IP address rather than the hostname, if you see the site then bindings are configured to allow any hostname.

Password Reset Poison

When a user resets their password, a URL is created containing the password reset token that the user then clicks to configure a new password. This URL is built using the **ApplicationUrl** and can therefore be controlled by the attacker, the code below is called when the user resets their password, the “*Current.RuntimeState.ApplicationUrl*” variable contains the attacker controlled URL

File: Umbraco.Web\Editors\AuthenticationController.cs (see lines 542 & 546 example code taken from Umbraco 8)

```
private string ConstructCallbackUrl(int userId, string code){
    //.. truncated for brevity..

    var applicationUri = Current.RuntimeState.ApplicationUrl;
    var callbackUri = new Uri(applicationUri, action);
    return callbackUri.ToString();
```

A quick and easy was to exploit this flaw is to edit your local host file and add an alternative entry for the target site. On Windows, this can be found in ‘*c:\windows\system32\drivers\etc\hosts*‘ (or /etc/hosts on

This website uses cookies to improve user experience. By using our website you consent to the storage of cookies from this website.
I agree

https://appcheck-ng.com/ was used. Umbraco reports the new Application URL within the log as shown in the screenshot below:

To test that the attack has been successful in persistently modifying the ApplicationUrl, reset your password and observe the URL used in the password reset URL:

Detection

AppCheck has included detection for this flaw since June 2021, prior to this AppCheck was detecting this flaw as “Out-of-Band Service Interaction (HTTP)”.

Solution

Partial Fix:

A partial fix has been deployed in Umbraco version 9.2.0 by the vendor that includes the following improvements:

- Password reset and user invites no longer use the cached ApplicationUrl. If no “UmbracoApplicationUrl” is configured, the value is enumerated again to use the hostname of the request invoking the password reset.

- A health check process now warns the administrator that the UmbracoApplicationUrl is not configured and recommends they do so. Once set, none of the flaws described in this post are exploitable.

Complete Fix:

Whilst the above improves the situation and removes the most critical aspect of this vulnerability, there are still some areas that remain vulnerable, such as;

- The password reset process could be invoked on behalf of the user with a malicious hostname set. The URL to reset the password is poisoned as before, however the user receives the email unexpectedly which would lower the likelihood of a successful attack (CVE-2022-22691).

- Some components are not covered by the fix such as; “Content Notifications, Healthcheck Notifications, and the Keep-Alive task”

A more comprehensive fix, available in all versions, is to specifically set the *UmbracoApplicationUrl* within the Umbraco Configuration. See Umbraco Settings Docs

This website uses cookies to improve user experience. By using our website you consent to the storage of cookies from this website.

I agree

21/07/2021: Umbraco confirms they can reproduce the issue

Discussions ongoing regarding the vulnerability and its pre-requisite configuration.

06/12/2021: Chased for a resolution

06/01/2022: Patch released, however release notes did not state a security flaw had been resolved.

13/01/2022: CVE's Assigned: CVE-2022-22690 & CVE-2022-22691

18/01/2022: Advisory Published

In line with MIRE guidelines, two CVE's were allocated. CVE-2022-22690 is used to track the vulnerability allowing the ApplicationUrl to be persistently overwritten. CVE-2022-22691 is used to track the ability to overwrite the password reset URL via the host header.

Get started with Appcheck

No software to download or install.
Contact us or call us 0113 887 8380

Start your free trial

POPULAR

DNS Security

The New OpenSSL Critical Vulnerability - Early Information and Detections

File Upload Vulnerabilities

RECENT ARTICLES

DNS Security

The New OpenSSL Critical Vulnerability - Early Information and Detections

File Upload Vulnerabilities

RELATED

This website uses cookies to improve user experience. By using our website you consent to the storage of cookies from this website.

I agree

AppCheck
ACCURACY IS EVERYTHING

Company

About Us

AppCheck Privacy Policy

Cookie Policy

Compliance Information

Existing Customer?

Keep in touch

email

I agree to receive information and commercial offers about AppCheck Ltd.

AppCheck Ltd. is a company registered in England and Wales with company number 06888174

Services

Web Application Scanner

Dynamic Application Security Testing (DAST)

Automated Penetration Testing

External Vulnerability Scanner

Resources

Brochure

OWASP Top 10

Sample Report

Free Trial

Book a Demo

This website uses cookies to improve user experience. By using our website you consent to the storage of cookies from this website.

I agree