**Weak Public/Private Keys**

The public key is shared with the Service Provider, which is used to verify the SAML response and then log the user in.

If an attacker can successfully change the name ID in the SAML response, they can log in as that user due to a lack of signature checking. To validate the signature, the X.509 public certificate of the Identity Provider (IdP) is required.

**Note:** You can validate the signature in three different ways; the one above is the most common.

The private key is used to sign SAML messages. As it says in the name, a private key is supposed to remain private.

**Identifying Weak Keys**

Firstly, Navigate to http://sp1.htb.net and log in with the credentials shared in the SAML section. You'll see a hyperlink. Click it, and multiple requests will start coming through. Just forward those until you click login (after entering credentials)

A screenshot of a computer

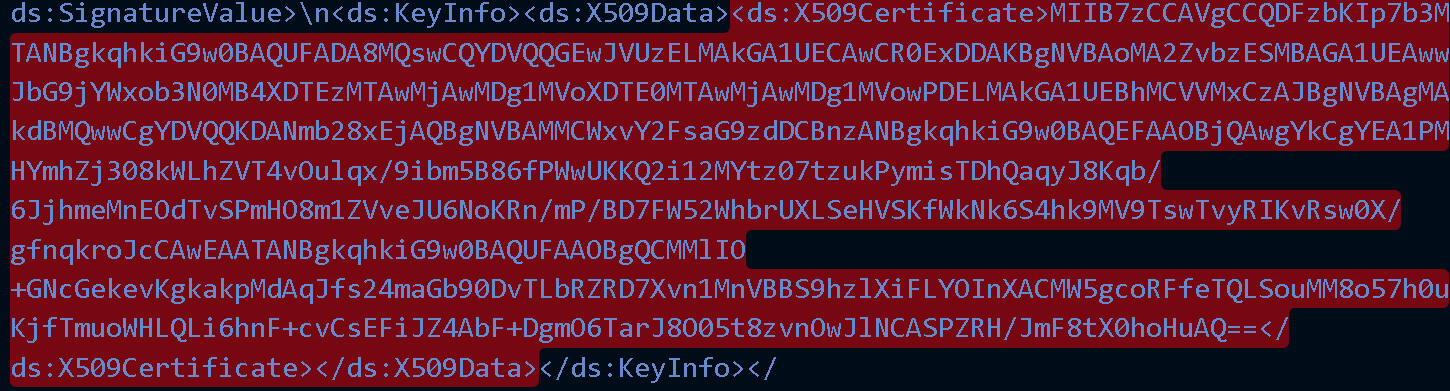
Description automatically generated

A screenshot of a computer

Description automatically generated

Send the request to Burp Repeater. This is the response, so copy the SAMLResponse payload. We need to URL decode the response body and then Base64 decode it.

Looking through the Base64 encoded values we come across this string:



Some private keys use a weak encryption algorithm, which was broken years ago, and all of its private keys have been published. So, we may be able to find the matching private key from its public key.

Let's copy the last part, DgmO6TarJ8O0...SNIP...tX0hoHuAQ== of the public key and perform some OSINT. The first thing to try is searching the string in Google:DgmO6TarJ8O0...SNIP...tX0hoHuAQ==.

A screenshot of a search engine

Description automatically generated

A screenshot of a computer screen

Description automatically generated

As we thought, let's try to find the private key.

Code: rsa

-----BEGIN RSA PRIVATE KEY-----

MIICXgIBAAKBgQDU8wdiaFmPfTyRYuFlVPi866WrH/2JubkHzp89bBQopDaLXYxi

3PTu3O6Q/KaKxMOFBqrInwqpv/omOGZ4ycQ51O9I+Yc7ybVlW94lTo2gpGf+Y/8E

<SNIP>

vBkm0F5yXKaYtoiiDMzlOQJADqmEwXl0D72ZG/2KDg8b4QZEmC9i5gidpQwJXUc6

hU+IVQoLxRq0fBib/36K9tcrrO5Ba4iEvDcNY+D8yGbUtA==

-----END RSA PRIVATE KEY-----

Via: [samlidptest](https://github.com/rstudio/crewjam-saml/blob/master/samlidp/samlidp_test.go)

A screenshot of a computer screen

Description automatically generated

**Attacking Weak Keys**

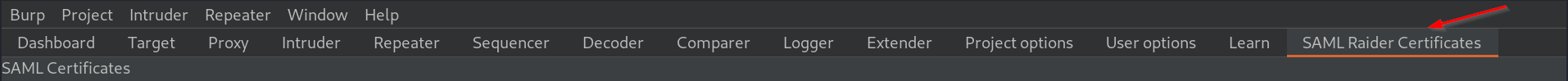
Now we've got everything we need. As we know, private RSA keys are kept private as secret keys. However, some private keys may be found online from their public keys!  
Let's continue our exploitation. For this, we'll be using a Burp Suite extension called [SAMLRaider](https://github.com/PortSwigger/saml-raider#:~:text=Start%20the%20Burp%20Suite%20and%20click%20at%20the,hit%20the%20Install%20button%20to%20install%20our%20extension)

You may install this extension from the BApp Store in Burp's Extender tab or may download the .jar file onto your attack host, then open Burp Suite and navigate to Extender -> Extensions -> Add and select the .jar file you downloaded.

A screenshot of a computer

Description automatically generated

Once done, click Next.



Now we need to add the private and public keys into their own individual files so we can load them into the SAMLRaider extension.

Navigate to https://github.com/rstudio/crewjam-saml/blob/master/samlidp/samlidp\_test.go and copy the private key and store it in a file named private.pem and the certificate below into a file called pub.crt

A screenshot of a computer program

Description automatically generated

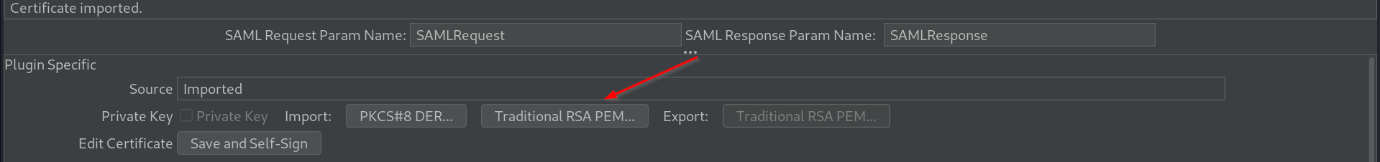
Create a directory at /tmp/certs, and add the files into the certs directory. Then, click Import in the Brup Suite SAML extension:

A black background with white text

Description automatically generated

Navigate to /tmp or the working directory and load the pub.crt file.

Once imported, click the imported cert, navigate below, and click the Traditional RSA Pem button.



Navigate to /tmp or the working directory and load the private.pem file.

Now it should look like this:

A grey rectangular object with white text

Description automatically generated

Next, we'll need the decoded SAML Response body values.

Code: xml

<saml:Attribute Name="username" NameFormat="urn:oasis:names:tc:SAML:2.0:attrname-format:uri"><saml:AttributeValue xsi:type="xs:string">admin</saml:AttributeValue>

We'll be changing the username SAMLAttribute value to hackme.

Navigate back to Burp Repeater, where the request was saved.

A screenshot of a computer screen

Description automatically generated

Load the SAMLRaider extension, and let's try to exploit the weakness here.

We'll first need to check whether or not the private key was loaded. As you can see, we are in the Repeater tab and can confirm the key was loaded successfully; thus, we can make requests or manipulate things as we please.

A screenshot of a computer

Description automatically generated

We need to change the username value, as shown above. However, this time change the string from admin or jake to hackme.

A screenshot of a computer

Description automatically generated

Send the request but do not click Follow redirection. The status code returned will be a 302.

A screenshot of a computer

Description automatically generated

As you can see, it's attempting to redirect us back to the root web directory. We do not want this.

We now need to Re-Sign the Assertion. We can do this because we have the private/public key. If we didn't have these, we wouldn't be able to sign the messages. We're signing it because there's been a change, and as you have seen, when we sent the first request, it triggered a 302 to / because the signature realized the request was modified.

A black rectangular object with white dots

Description automatically generated

Now send the request.

A screenshot of a computer

Description automatically generated

Copy the JWT from the response.

Navigate to [jwt.io](https://jwt.io).

A screenshot of a computer code

Description automatically generated

As we can see, we have obtained an authenticated JWT token by getting the user's private key.

In some cases, we may be able to perform the same attack without even having the private key if the SAML was not securely configured. Once we capture the request, we can choose Send certificate to SAML Raider Certificates, then go to the extension and select Save and Self-sign, and the self-signature may also work. In the next section, we will see a similar attack.