# DVWS XML API Lab Guide for WAF

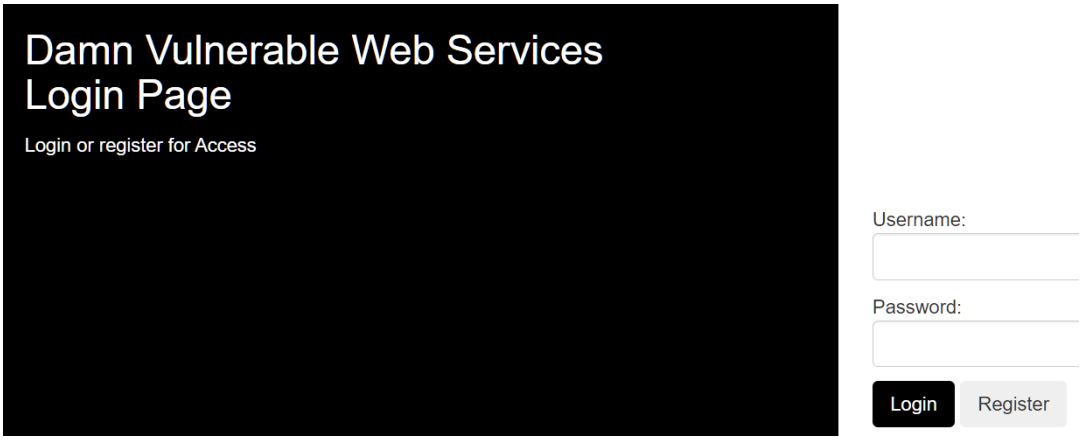
April 2021

1. You will need curl and a command prompt for this lab, so make sure you have both of those things. If you have a Mac, Linux box, or a Windows 10, you most likely have it. You can also use Azure Cloud Shell or AWS Cloud Shell. Make sure copy and paste works because you are going to be copying and pasting. Copy and Paste the following to test it works:

   **curl –version**

```
C:\Users\brett>curl.exe --version
curl 7.55.1 (Windows) libcurl/7.55.1 WinSSL
Release-Date: 2017-11-14, security patched: 2019-11-05
Protocols: dict file ftp ftps http https imap imaps pop3 pop3s smtp smtps telnet tftp
Features: AsynchDNS IPv6 Largefile SSPI Kerberos SPNEGO NTLM SSL

C:\Users\brett>
```

2. Login to your WAF, then create a service listening on 443 with a server named dvws.cudathon.com listening on 80.
3. Browse to the service and make sure it works. You can login as test/test just to check it out, but there is nothing to do for this step other than make sure it works.




4. Also do this via curl from your command line, make sure it loads

   **curl –k https://waf.brett1.com**

```
]$ curl -k https://waf.brett1.com/
        post.success(function (data, status) {
            if (data.status == 201) {
            $scope.DataResponse = data.user + ' created success
             } else if (data.status == 409) {
            $scope.DataResponse = data;
             }
        });

        post.error(function (data, status) {
                $scope.DataResponse = data;
            });
        }

    });
```

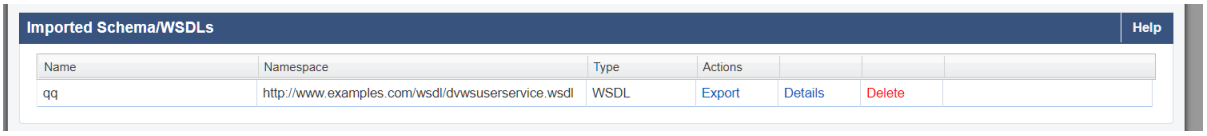
5. Download the DVWS WSDL file from https://raw.githubusercontent.com/snoopysecurity/dvws-node/master/soapserver/dvwsuserservice.wsdl to your computer

6. As you can see, at the bottom of this file, the location is set to waf.brett1.com. Change it to match you service fqdn

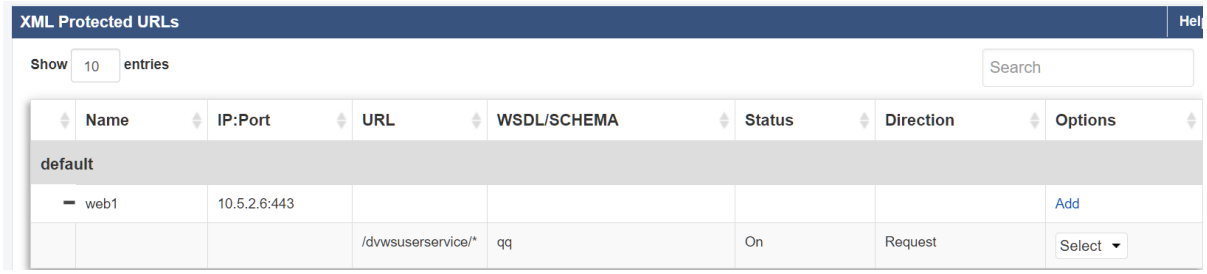
dvwsuserservice.wsdl ×

```
<definitions name="UserService"
   targetNamespace="http://www.examples.com/wsdl/dvwsuserservice.wsdl"
   xmlns="http://schemas.xmlsoap.org/wsdl/"
   xmlns:soap="http://schemas.xmlsoap.org/wsdl/soap/"
   xmlns:tns="http://www.examples.com/wsdl/dvwsuserservice.wsdl"
   xmlns:xsd="http://www.w3.org/2001/XMLSchema">

   <message name="UsernameRequest">
      <part name="username" type="xsd:string"/>
   </message>

   <message name="UsernameResponse">
      <part name="username" type="xsd:string"/>
   </message>

   <portType name="Username_PortType">
      <operation name="Username">
         <input message="tns:UsernameRequest"/>
         <output message="tns:UsernameResponse"/>
      </operation>
   </portType>

   <binding name="Username_Binding" type="tns:Username_PortType">
      <soap:binding style="rpc"
         transport="http://schemas.xmlsoap.org/soap/http"/>
      <operation name="Username">
         <soap:operation soapAction="Username"/>
         <input>
            <soap:body
               encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
               namespace="urn:examples:usernameservice"
               use="encoded"/>
         </input>

         <output>
            <soap:body
               encodingStyle="http://schemas.xmlsoap.org/soap/encoding/"
               namespace="urn:examples:usernameservice"
               use="encoded"/>
         </output>
      </operation>
   </binding>

   <service name="User_Service">
      <documentation>WSDL File for DVWS User Service</documentation>
      <port binding="tns:Username_Binding" name="Username_Port">
         <soap:address
            location="https://waf.brett1.com/dvwsuserservice/" />
      </port>
   </service>
</definitions>
```

7. WAF->Websites->XML Validations->Import Schema/WSDL, select WSDL, give it a name, and choose the file you downloaded, and click import. Namespace is not needed just leave it blank.




8. WAF->Websites->XML Validations->XML Protected URLs add the WSDL to the /dvwsuserservice/* url and wildcard * host as shown




9. Download the request.xml file from here, and save it where you are running curl from, because you are going to need it in the next step https://raw.githubusercontent.com/bwolmarans/legendary-disco/main/request.xml

```
<soapenv:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/"
   <soapenv:Header/>
   <soapenv:Body>
      <urn:Username>
         <username xsi:type="xsd:string">gero et</username>
      </urn:Username>
   </soapenv:Body>
</soapenv:Envelope>
```

10. Do a SOAP request using curl by copy and pasting the following:

**curl -X POST -H "Content-Type: text/xml" -H 'SOAPAction: "Username"' --data-binary @request.xml https://waf.brett1.com/dvwsuserservice/**

and validate you get a response User Not Found

```
[centos@ip-10-0-1-234 partnerlab]$ curl -X POST -H "Content-Type: text/xml" -H 'SOAPAction: "Username"' --data-binary @request.xml https://waf.brett1.com/dvwsuserservice/
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<soapenv:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soapenv="http://schemas.xmlsoap.org/soap/envelope/" xmlns
:urn="urn:examples:usernameservice">
  <soapenv:Header/>
  <soapenv:Body>
    <urn:UsernameResponse soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
      <username xsi:type="xsd:string">User Not Found:gero et</username>
    </urn:UsernameResponse>
  </soapenv:Body>
[centos@ip-10-0-1-234 partnerlab]$
```

11. Download the misconfigured.xml file from here and save it where you can access it using curl https://raw.githubusercontent.com/bwolmarans/legendary-disco/main/misconfigured.xml This XML is misconfigured because it has petname instead of username, which is in violation of the XML Schema
12. Do a SOAP request like this by copy and pasting the following:

**curl -X POST -H "Content-Type: text/xml" -H 'SOAPAction: "Username"' --data-binary @misconfigured.xml https://waf.brett1.com/dvwsuserservice/**

and make sure the WAF blocks it because it enforces the schema

| Time | Event Details | Client Details | Attack Details | Actions |
|---|---|---|---|---|
| DENIED | URL /dvwsuserservice/ | | | |
| Time 10:13:14.160 | Service IP:Port 10.5.2.6:443 | Client IP 18.232.46.9 | Attack Name Envelope Does Not Conform to S | Fix Details |
| Date 2021-04-27 | Service Name web1 | Country US | Attack Detail The envelope does not conform t | |
| ID 179145214af-e16b1c5d | Protocol TLSv1.2 | Method POST | Rule global<br>Policy Name default | |

```
[centos@ip-10-0-1-234 partnerlab]$ curl -X POST -H "Content-Type: text/xml" -H 'SOAPAction: "Username"' --data-binary @misconfigured.xml https://waf.brett1.com/dvwsuserservice/
<?xml version="1.0" encoding="utf-8"?>
  <soap:Envelope xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/">
    <soap:Body>
      <soap:Fault>
        <faultcode>soap:Server</faultcode>
        <faultstring>Server was unable to process request. --&gt; The envelope does not conform to the SOAP schema located at http://schemas.xmlsoap.org/soap/envelope/. </faultstrin
g>
        <detail />
      </soap:Fault>
    </soap:Body>
  </soap:Envelope>
[centos@ip-10-0-1-234 partnerlab]$ cat misconfigured.xml
```

13. Try the same misconfigured SOAP request as before on the origin server and you will see it causes unwanted errors and information leakage showing details of node backend on the origin server

```
[centos@ip-10-0-1-234 partnerlab]$ curl -X POST -H "Content-Type: text/xml" -H 'SOAPAction: "Username"' --data-binary @misconfigured.xml http://3.238.162.130/dvwsuserservice/
<!DOCTYPE html>
<html lang="en">
<head>
<meta charset="utf-8">
<title>Error</title>
</head>
<body>
<pre>TypeError: Cannot read property &#39;text&#39; of undefined<br>    at /home/dvws-node/soapserver/dvwsuserservice.js:54:29<br>    at Layer.handle [as handle
_request] (/home/dvws-node/node_modules/express/lib/router/layer.js:95:5)<br>    at next (/home/dvws-node/node_modules/express/lib/router/route.js:137:13)<br>   &nbs
p;at Route.dispatch (/home/dvws-node/node_modules/express/lib/router/route.js:112:3)<br>    at Layer.handle [as handle_request] (/home/dvws-node/node_modules/express/lib/
router/layer.js:95:5)<br>    at /home/dvws-node/node_modules/express/lib/router/index.js:281:22<br>    at Function.process_params (/home/dvws-node/node_modules/
express/lib/router/index.js:335:12)<br>    at next (/home/dvws-node/node_modules/express/lib/router/index.js:275:10)<br>    at timeLogStart (/home/dvws-node/soa
pserver/dvwsuserservice.js:19:5)<br>    at Layer.handle [as handle_request] (/home/dvws-node/node_modules/express/lib/router/layer.js:95:5)<br>    at trim_prefi
x (/home/dvws-node/node_modules/express/lib/router/index.js:317:13)<br>    at /home/dvws-node/node_modules/express/lib/router/index.js:284:7<br>    at Function.
process_params (/home/dvws-node/node_modules/express/lib/router/index.js:335:12)<br>    at next (/home/dvws-node/node_modules/express/lib/router/index.js:275:10)<br> &nbs
p;  at /home/dvws-node/node_modules/body-parser/lib/read.js:130:5<br>    at invokeCallback (/home/dvws-node/node_modules/raw-body/index.js:224:16)</pre>
</body>
</html>
[centos@ip-10-0-1-234 partnerlab]$
```

14. Download the exploit.xml file from [https://raw.githubusercontent.com/bwolmarans/legendary-disco/main/exploit.xml](https://raw.githubusercontent.com/bwolmarans/legendary-disco/main/exploit.xml) and save it where you can access it using curl. Try it against the origin server like this, it will dump the passwd file, by copy and pasting the following:

**curl -X POST -H "Content-Type: text/xml" -H 'SOAPAction: "Username"' --data-binary @exploit.xml [https://waf.brett1.com/dvwsuserservice/](https://waf.brett1.com/dvwsuserservice/)**

```
[centos@ip-10-0-1-234 partnerlab]$ curl -X POST -H "Content-Type: text/xml" -H 'SOAPAction: "Username"' --data-binary @exploit.xml http://3.238.162.130/dvwsuserservice/
<?xml version="1.0" encoding="UTF-8" standalone="yes"?>
<soapenv:Envelope xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema" xmlns:soapenv="http://schemas.xmlsoap.org/soap/envel
:urn="urn:examples:usernameservice">
  <soapenv:Header/>
  <soapenv:Body>
    <urn:UsernameResponse soapenv:encodingStyle="http://schemas.xmlsoap.org/soap/encoding/">
      <username xsi:type="xsd:string">User Not Found:root:x:0:0:root:/root:/bin/bash
daemon:x:1:1:daemon:/usr/sbin:/usr/sbin/nologin
bin:x:2:2:bin:/bin:/usr/sbin/nologin
sys:x:3:3:sys:/dev:/usr/sbin/nologin
sync:x:4:65534:sync:/bin:/bin/sync
games:x:5:60:games:/usr/games:/usr/sbin/nologin
man:x:6:12:man:/var/cache/man:/usr/sbin/nologin
lp:x:7:7:lp:/var/spool/lpd:/usr/sbin/nologin
mail:x:8:8:mail:/var/mail:/usr/sbin/nologin
news:x:9:9:news:/var/spool/news:/usr/sbin/nologin
uucp:x:10:10:uucp:/var/spool/uucp:/usr/sbin/nologin
proxy:x:13:13:proxy:/bin:/usr/sbin/nologin
www-data:x:33:33:www-data:/var/www:/usr/sbin/nologin
backup:x:34:34:backup:/var/backups:/usr/sbin/nologin
list:x:38:38:Mailing List Manager:/var/list:/usr/sbin/nologin
irc:x:39:39:ircd:/var/run/ircd:/usr/sbin/nologin
gnats:x:41:41:Gnats Bug-Reporting System (admin):/var/lib/gnats:/usr/sbin/nologin
nobody:x:65534:65534:nobody:/nonexistent:/usr/sbin/nologin
_apt:x:100:65534::/nonexistent:/bin/false
node:x:1000:1000::/home/node:/bin/bash
</username>
    </urn:UsernameResponse>
  </soapenv:Body>
</soapenv:Envelope>[centos@ip-10-0-1-234 partnerlab]$
```

15. Now try it against the WAF and see the WAF blocks it because of the attack embedded in it