

# Web Services – Attacks and Defense

*Information Gathering Methods: Footprints, Discovery & Fingerprints*


## Abstract

Web Services is growing at a rapid rate and bringing into focus, new security issues in the web security landscape. How do we start assessing web services deployed at any corporate location? That is the fundamental question and once again it all starts with information gathering. UDDI, WSDL and SOAP are three cornerstones of this technology and they can be powerful tools for information gathering. Universal Business Registry (UBR) can help in footprinting using UDDI. UBR and technology fingerprinting can be used to perform discovery of web services. The scope in this paper is limited to only the first phase, namely the Web Services Information Gathering Phase. The entire methodology for web services information gathering is covered in this paper. The next two phases of the Assessment methodology are enumeration and defining attack vectors, both extensive topics too. These will be taken up in later papers.



**Shreeraj Shah**

Co-Author: "Web Hacking: Attacks and Defense" (Addison Wesley, 2002) and published several advisories on security flaws.

*shreeraj@net-square.com*

*http://www.net-square.com*

**Index**



## Acknowledgement

*Lyra Fernandes* for her help on documentation.



### 1.1 Background

The growth of Web services technology in the last five years has been phenomenal. A lot of new technologies are emerging, which support web services and that is providing a momentum to adoption of these technologies in the market. Gartner and other research groups are suggesting to companies to adopt these technologies or be left out of the competition. One of the recent surveys suggests that web service offerings are going to skyrocket to a staggering $34 billion by 2007 from the current $1.6 billion.

An increase in the growth of web services technology and an even faster adoption by companies looking for that extra business edge, raises security concerns. New toolkits, vulnerabilities and exploits are emerging that are specifically targeting web services. Older web application attacks no longer work in this newly developed application security framework incorporating application layer content filtering. Web services are still confusing because of the many protocols in use and that is opening up a whole new range of vulnerabilities and security issues.

### 1.2 Problem Area

A traditional security framework has essentially three levels – Operating systems, Services and Application level. Web Services adds a new level in the framework and that is the business application layer. Firewalls are effective at blocking traffic directed at the operating system and services level but are unable to block web application level attacks since traffic on ports 80 and 443 is legitimate.

An application layer firewall provides a layer of content filtering that blocks attacks such as *SQL injection* and *Parameter Tampering*, though not without some limitations. New attacks aimed at web services cannot be stopped without proper defense. This is the reason web service is emerging as security concern and problem area. Some of the new emerging challenges in web services security are:

1. Footprinting web services
2. Gathering information for web services
3. Identifying technologies
4. Discovering endpoints

### 1.3 Approach

Web Services assessment can be done in following steps:

a.) Web Services information gathering
b.) Web Services enumeration
c.) Web Services attack vector and defense strategies

This paper is written to address (a) in the above case and will facilitate understanding of the entire web services information gathering exercise, the methodical approach and organization of information. This approach helps in the assessment of various web applications using web services. Areas (b) and (c) will be addressed in different research papers.

# 2. Web Service Information Gathering Methodology

## 2.1 Basics

Web Services has three critical building blocks – UDDI, WSDL and SOAP. There are two players – *Web services consumer* and *Web services supplier*. All interaction between these two players is carried out using these three building blocks. There is a third intermediate player facilitating communication between the consumer and supplier, referred to as Universal Business Registry (UBR).

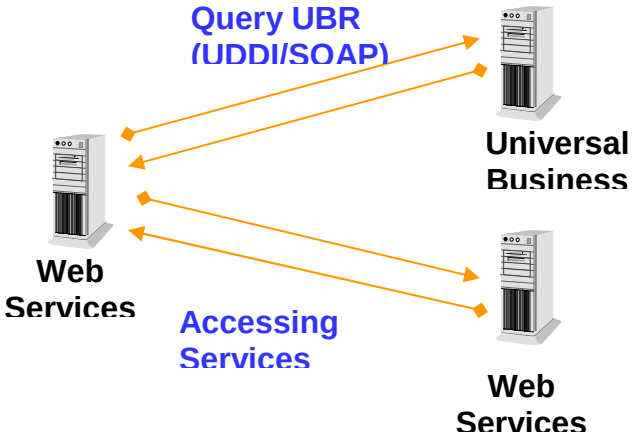



**Figure 1 : Players and Actions**

## 2.2 Actors, Protocols and Interaction

As shown in Figure 1, the following process takes place between the different entities:

1.) Web Services Consumer queries the UBR and looks for services as per requirements.
2.) UBR supplies the list of available services. The Web Services Consumer chooses one or more available services.
3.) Web Services Consumer requests for an access point or end point for these services. UBR supplies this information.
4.) Web Services Consumer approaches the Web Services Supplier's Host/IP address and starts accessing service.

Explained below are the protocols are used during each step of the entire process:

***1.) UDDI (Universal Description, Discovery and Integration)*** – Steps (1) to (3) occur on this protocol. The SOAP messaging system runs over this protocol.

***2.) WSDL (Web Services Definition Language)*** – Step (4) in the above process uses this method to understand how web services are deployed and how these services can be accessed over the network. Once again this works over HTTP/HTTPS on the internet.

***3.) SOAP (Simple Object Access Protocol)*** – It is a communication protocol which sits on top of the HTTP/HTTPS protocol and a few others. It acts as an underlying communication messaging system throughout the above process.

### 2.3 Steps of Information Gathering

There are various methods available for gathering information with zero-level knowledge about a web service. This zero-level knowledge may be an organization name or employee email address. In traditional methods, this search for information would begin with a "whois" query, followed by attempts to obtain entire IP address ranges. In recent years, the addition of new methods of querying a search engine database for information has contributed to an amazing depth of information that can be gleaned from repositories.

They key question is: In a similar situation, how do we start with web services?
For our example, let us assume that the only bit of information that we have is the name of the company. We seek information on the kind of web services this company offers to close associates and clients. Remember, except for the company name, we don't have very much to go on. Here's the methodology by which we can go about collecting the information we require.

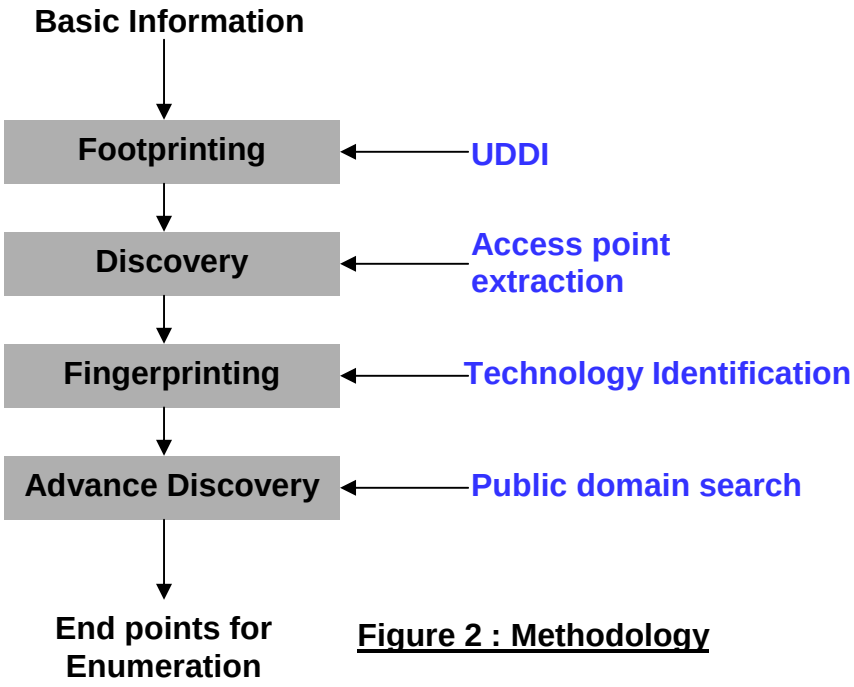



**Figure 2 : Methodology**

As shown in Figure 2, information gathering can be done using the following steps:

1. Web Services Footprinting
2. Web Services Discovery
3. Web Services Technology fingerprinting
4. Advance discovery on public domain

The steps listed above form the core methodology for achieving our objectives and to identify threats associated with information leakage for web services. This is the crucial part of assessment of web applications using web services. Each is explained in detail in subsequent sections.

### 3.1 Footprinting

As discussed in the preceding section, Universal Business Registry (UBR) is a major source of information for web services. It is the place where businesses register web services and consumer search for services – a public registry – and is very similar in function to a "whois" server. Queries sent to UBR result in responses being sent back with the required information. UBR runs on UDDI specifications and SOAP.

Web services can be registered on UBR using one of the following structures:

1. Business Entity
2. Business Service
3. Binding Template
4. Technical Model (tModel)

UBRs are created by various companies such as Microsoft, IBM, SAP etc. These companies replicate their data with one another. There is set of APIs which can help in querying this registry. We can use each of these APIs to query UBR for specific information. For example if we are looking for information on "amazon". We can footprint the company using all these APIs and see what information can be extracted. These APIs work on HTTP/HTTPS with SOAP. The WSDL specification is already published and can be used by *inquire APIs* to extract the requisite information.

Different sets of toolkits and SDKs are published and developed by various companies such as Microsoft, Sun etc. These toolkits can be used to write programs to enumerate or just invoke the APIs using SOAP on simple TCP clients such as netcat. We illustrate below a simple way to generate a request using a simple TCP client and send this request on the network. As shown in the screenshot below, http://uddi.microsoft.com/ is our UBR for information footprinting.

### 3.2 Footprinting on Business Name

API: find_business

***Request:***

```
POST /inquire HTTP/1.0
Content-Type: text/xml; charset=utf-8
SOAPAction: ""
Host: uddi.microsoft.com
Content-Length: 229

<?xml version="1.0" encoding="UTF-8" ?><Envelope
xmlns="http://schemas.xmlsoap.org/soap/envelope/"><Body><find_business generic="2.0"
maxRows="100" xmlns="urn:uddi-
org:api_v2"><name>amazon</name></find_business></Body></Envelope>
```

***Response:***

```
HTTP/1.1 200 OK
Date: Tue, 28 Sep 2004 09:53:53 GMT
Server: Microsoft-IIS/6.0
X-Powered-By: ASP.NET
X-AspNet-Version: 1.1.4322
Cache-Control: private, max-age=0
Content-Type: text/xml; charset=utf-8
Content-Length: 1339

<?xml version="1.0" encoding="utf-8"?><soap:Envelope
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"><soap:Body><businessList generic="2.0"
operator="Microsoft Corporation" truncated="false" xmlns="urn:uddi-
org:api_v2"><businessInfos><businessInfo businessKey="bfb9dc23-adec-4f73-bd5f-
5545abaeaa1b"><name xml:lang="en-us">Amazon Web Services for Testing</name><description
xml:lang="ko">Amazon Web Services 2.0 - We now offer software developers the opportunity to
integrate Amazon.com</description><serviceInfos><serviceInfo serviceKey="41213238-1b33-40f4-
8756-c89cc3125ecc" businessKey="bfb9dc23-adec-4f73-bd5f-5545abaeaa1b"><name xml:lang="en-
us">Amazon Web Services 2.0</name></serviceInfo></serviceInfos></businessInfo><businessInfo
businessKey="18b7fde2-d15c-437c-8877-ebec8216d0f5"><name
xml:lang="en">Amazon.com</name><description xml:lang="en">E-commerce website and
platform for finding, discovering, and buying products
online.</description><serviceInfos><serviceInfo serviceKey="ba6d9d56-ea3f-4263-a95a-
eeb17e5910db" businessKey="18b7fde2-d15c-437c-8877-ebec8216d0f5"><name
xml:lang="en">Amazon.com Web
Services</name></serviceInfo></serviceInfos></businessInfo></businessInfos></businessList></so
ap:Body></soap:Envelope>
```

Analyzing the above response, we observe that the nodes give us information on each business registered with a business name containing the string “amazon”. From the above information block we can identify services associated with the business as well. The following two business names and their corresponding business registry keys were found.

Amazon Web Services for Testing <bfb9dc23-adec-4f73-bd5f-5545abaeaa1b>
Amazon.com <18b7fde2-d15c-437c-8877-ebec8216d0f5>

### 3.3 Footprinting on Service Name

API: find_service

***Request:***

```
POST /inquire HTTP/1.0
Content-Type: text/xml; charset=utf-8
SOAPAction: ""
Host: uddi.microsoft.com
Content-Length: 213

<?xml version="1.0" encoding="UTF-8" ?><Envelope
xmlns="http://schemas.xmlsoap.org/soap/envelope/"><Body><find_service eneric="2.0"
xmlns="urn:uddi-rg:api_v2"><name>amazon</name></find_service></Body></Envelope>
```

***Response:***

```
HTTP/1.1 200 OK
Date: Tue, 28 Sep 2004 10:07:42 GMT
Server: Microsoft-IIS/6.0
X-Powered-By: ASP.NET
X-AspNet-Version: 1.1.4322
Cache-Control: private, max-age=0
Content-Type: text/xml; charset=utf-8
Content-Length: 1272

<?xml version="1.0" encoding="utf-8"?><soap:Envelope
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"><soap:Body><serviceList generic="2.0"
operator="Microsoft Corporation" truncated="false" xmlns="urn:uddi-
org:api_v2"><serviceInfos><serviceInfo serviceKey="6ec464e0-2f8d-4daf-b4dd-5dd4ba9dc8f3"
businessKey="914374fb-f10f-4634-b8ef-c9e34e8a0ee5"><name xml:lang="en-us">Amazon
Research Pane</name></serviceInfo><serviceInfo serviceKey="41213238-1b33-40f4-8756-
c89cc3125ecc" businessKey="bfb9dc23-adec-4f73-bd5f-5545abaeaa1b"><name xml:lang="en-
us">Amazon Web Services 2.0</name></serviceInfo><serviceInfo serviceKey="ba6d9d56-ea3f-
4263-a95a-eeb17e5910db" businessKey="18b7fde2-d15c-437c-8877-ebec8216d0f5"><name
xml:lang="en">Amazon.com Web Services</name></serviceInfo><serviceInfo
serviceKey="bc82a008-5e4e-4c0c-8dba-c5e4e268fe12" businessKey="18785586-295e-448a-
b759-ebb44a049f21"><name
xml:lang="en">AmazonBookPrice</name></serviceInfo><serviceInfo serviceKey="8faa80ea-
42dd-4c0d-8070-999ce0455930" businessKey="ee41518b-bf99-4a66-9e9e-c33c4c43db5a"><name
xml:lang="en">AmazonBookPrice</name></serviceInfo></serviceInfos></serviceList></soap:Bo
dy></soap:Envelope>
```

In the above response, the following node gives us information on each of the services registered with a service name containing the string “amazon”. From the above information block we can identify services. The following service names and corresponding service keys were found.

Amazon Research Pane <6ec464e0-2f8d-4daf-b4dd-5dd4ba9dc8f3>
Amazon Web Services 2.0 <41213238-1b33-40f4-8756-c89cc3125ecc>
Amazon.com Web Services <ba6d9d56-ea3f-4263-a95a-eeb17e5910db>
AmazonBookPrice <bc82a008-5e4e-4c0c-8dba-c5e4e268fe12>
AmazonBookPrice <8faa80ea-42dd-4c0d-8070-999ce0455930>

### 3.4 Footprinting on tModel

API: find_tModel

***Request:***

```
POST /inquire HTTP/1.0
Content-Type: text/xml; charset=utf-8
SOAPAction: ""
Host: uddi.microsoft.com
Content-Length: 211

<?xml version="1.0" encoding="UTF-8" ?><Envelope
xmlns="http://schemas.xmlsoap.org/soap/envelope/"><Body><find_tModel generic="2.0"
xmlns="urn:uddi-org:api_v2"><name>amazon</name></find_tModel></Body></Envelope>
```

***Response:***

```
HTTP/1.1 200 OK
Date: Tue, 28 Sep 2004 10:12:42 GMT
Server: Microsoft-IIS/6.0
X-Powered-By: ASP.NET
X-AspNet-Version: 1.1.4322
Cache-Control: private, max-age=0
Content-Type: text/xml; charset=utf-8
Content-Length: 516

<?xml version="1.0" encoding="utf-8"?><soap:Envelope
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"><soap:Body><tModelList generic="2.0"
operator="Microsoft Corporation" truncated="false" xmlns="urn:uddi-
org:api_v2"><tModelInfos><tModelInfo tModelKey="uuid:c5da9443-d058-4ede-9db1-
4f1d5deb805c"><name>Amazon Web Services 2.0 WSDL
File</name></tModelInfo></tModelInfos></tModelList></soap:Body></soap:Envelope>
```

In the above response, the following node gives us information on each tModel registered with tModel name containing the string “amazon”. From the above information block we can identify tModels. The following tModels and corresponding tModel keys were found.

Amazon Web Services 2.0 WSDL File <uuid:c5da9443-d058-4ede-9db1-4f1d5deb805c>

### 3.5 Meta characters and Tools

By using the above methods we can generate a tool to query more than one business registry nodes over the Internet and gather all requisite information. This information may be specific to business, service and tModel names. While supplying names, UDDI supports some meta characters as well. For example % means “starts from” – which means that if we are looking for names beginning with the word “amazon”, we can use “amazon%” as part of our query.

### 4.1 Discovery

The process of Footprinting makes available to us the list of registered services on UBR. The objective of discovery is to identify access points for each of these services. An access point contains a host/IP address for the services and resource location on the server. This access point acts as key component for information gathering and enumeration of web service information begins from here.

Once again, UBR is the source for this information as well. Discovery using the same protocol can be performed and UDDI APIs can then be used to extract that information. To get back to our example, during the footprinting phase we obtained registry keys for business, service and tModel. This key can be used to identify the access point associated with it.

### 4.2 Discovery on Business Name

To perform discovery based on business name we must first identify services for that business name.
For example, let us assume we are looking for services on the basis of the business name Amazon.com <18b7fde2-d15c-437c-8877-ebec8216d0f5>. In first step we will identify services for this business name.

API: find_service

***Request:***

```
POST /inquire HTTP/1.0
Content-Type: text/xml; charset=utf-8
SOAPAction: ""
Host: uddi.microsoft.com
Content-Length: 245

<?xml version="1.0" encoding="UTF-8" ?><Envelope
xmlns="http://schemas.xmlsoap.org/soap/envelope/"><Body><find_service
businessKey="18b7fde2-d15c-437c-8877-ebec8216d0f5" generic="2.0" xmlns="urn:uddi-
org:api_v2"></find_service></Body></Envelope>
```

***Response:***

```
HTTP/1.1 200 OK
Date: Wed, 13 Oct 2004 12:51:26 GMT
Server: Microsoft-IIS/6.0
X-Powered-By: ASP.NET
X-AspNet-Version: 1.1.4322
Cache-Control: private, max-age=0
Content-Type: text/xml; charset=utf-8
Content-Length: 573
```

```
<?xml version="1.0" encoding="utf-8"?><soap:Envelope
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"><soap:Body><serviceList generic="2.0"
operator="Microsoft Corporation" truncated="false" xmlns="urn:uddi-
org:api_v2"><serviceInfos><serviceInfo serviceKey="ba6d9d56-ea3f-4263-a95a-eeb17e5910db"
businessKey="18b7fde2-d15c-437c-8877-ebec8216d0f5"><name xml:lang="en">Amazon.com
Web
Services</name></serviceInfo></serviceInfos></serviceList></soap:Body></soap:Envelope>
```

### 4.3 Discovery on services

API: get_serviceDetail

In our footprinting section, we retrieved the service name Amazon.com Web Services <ba6d9d56-ea3f-4263-a95a-eeb17e5910db> The preceding section also explained how the same service key could be retrieved using just the business name. Now, using this service key we want to retrieve service detail.

***Request:***

```
POST /inquire HTTP/1.0
Content-Type: text/xml; charset=utf-8
SOAPAction: ""
Host: uddi.microsoft.com
Content-Length: 265

<?xml version="1.0" encoding="UTF-8" ?><Envelope
xmlns="http://schemas.xmlsoap.org/soap/envelope/"><Body><get_serviceDetail generic="2.0"
xmlns="urn:uddi-org:api_v2"><serviceKey>ba6d9d56-ea3f-4263-a95a-
eeb17e5910db</serviceKey></get_serviceDetail></Body></Envelope>
```

***Response:***

```
HTTP/1.1 200 OK
Date: Wed, 13 Oct 2004 12:47:35 GMT
Server: Microsoft-IIS/6.0
X-Powered-By: ASP.NET
X-AspNet-Version: 1.1.4322
Cache-Control: private, max-age=0
Content-Type: text/xml; charset=utf-8
Content-Length: 1275

<?xml version="1.0" encoding="utf-8"?><soap:Envelope
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"><soap:Body><serviceDetail generic="2.0"
operator="Microsoft Corporation" truncated="false" xmlns="urn:uddi-
org:api_v2"><businessService serviceKey="ba6d9d56-ea3f-4263-a95a-eeb17e5910db"
businessKey="18b7fde2-d15c-437c-8877-ebec8216d0f5"><name xml:lang="en">Amazon.com
Web Services</name><description xml:lang="en">Set of web services that allow developers to
create applications that consume Amazon.com core features. When tied to Amazon.com Associate
program, developers can earn a percentage of each transaction that Amazon.com fullfills.
```

Developers must have a token</description><bindingTemplates><bindingTemplate bindingKey="1d3cf316-6b47-430b-9b8b-277a6e321e33" serviceKey="ba6d9d56-ea3f-4263-a95a-eeb17e5910db"><description xml:lang="en">The WSDL file that allows developers to make use of Amazon.com features on their own site.</description><accessPoint URLType="http">http://soap.amazon.com/schemas/AmazonWebServices.wsdl</accessPoint><tModelInstanceDetails /></bindingTemplate></bindingTemplates></businessService></serviceDetail></soap:Body></soap:Envelope>

In the above response we have located the URL for this web service as our access point.
Discovery URL : http://soap.amazon.com/schemas/AmazonWebServices.wsdl

### 4.4 Discovery on tModel

API: get_tModelDetail
In our footprinting section, we retrieved tModel for Amazon Web Services 2.0 WSDL File <uuid:c5da9443-d058-4ede-9db1-4f1d5deb805c>

Now, with this tModel key we can send following request:

***Request:***

```
POST /inquire HTTP/1.0
Content-Type: text/xml; charset=utf-8
SOAPAction: ""
Host: uddi.microsoft.com
Content-Length: 389

<?xml version="1.0" encoding="utf-8"?><soap:Envelope
xmlns:soap="http://schemas.xmlsoap.org/soap/envelope/"
xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance"
xmlns:xsd="http://www.w3.org/2001/XMLSchema"><soap:Body><get_tModelDetail generic="2.0"
xmlns="urn:uddi-org:api_v2"><tModelKey>uuid:c5da9443-d058-4ede-9db1-
4f1d5deb805c</tModelKey></get_tModelDetail></soap:Body></soap:Envelope>
```

***Response:***

```
HTTP/1.1 200 OK
Connection: close
Date: Fri, 15 Oct 2004 11:06:54 GMT
Server: Microsoft-IIS/6.0
X-Powered-By: ASP.NET
X-AspNet-Version: 1.1.4322
Cache-Control: private, max-age=0
Content-Type: text/xml; charset=utf-8
Content-Length: 788

<?xml version="1.0" encoding="utf-8"?><soap:Envelope xmlns:soap="http://schemas.
xmlsoap.org/soap/envelope/" xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance
" xmlns:xsd="http://www.w3.org/2001/XMLSchema"><soap:Body><tModelDetail generic=
"2.0" operator="Microsoft Corporation" truncated="false" xmlns="urn:uddi-org:api
_v2"><tModel tModelKey="uuid:c5da9443-d058-4ede-9db1-4f1d5deb805c" operator="Mic
rosoft Corporation" authorizedName="runtou"><name>Amazon Web Services 2.0 WSDL F
ile</name><description xml:lang="ko">Amazon Web Services 2.0 WSDL File</descript
ion><overviewDoc><description xml:lang="ko">Amazon Web Services 2.0</description
><overviewURL>http://soap.amazon.com/schemas2/AmazonWebServices.wsdl</overviewUR
L></overviewDoc></tModel></tModelDetail></soap:Body></soap:Envelope>
```

In the above response we have located a URL for this tModel as our access point.
Discovery URL : http://soap.amazon.com/schemas2/AmazonWebServices.wsdl

# 5. Web Services Technology fingerprinting

## 5.1 Technology fingerprinting or Identification

One of the challenges in the security space is to fingerprint technologies and gather information on each of the technologies. It can be an ongoing process and entails using or developing many methods. At this point and as part of this paper we will try to address the question: After obtaining a discovery URL what can we identify by just looking at the string of characters? We will address two technologies for web services - .Net and Java Web services running on Axis.

## 5.2 Technology fingerprinting using extensions

As an example, let us consider the following two discovery URLs:

1. http://example.com/customer/getinfo.asmx
2. http://example.com/supplier/sendinfo.jws

*asmx/jws extension*

This is part of .Net/J2EE frameworks resource for web services and web services can be developed/deployed using this type of resource. Hence, by just glancing at the set of characters containing the .asmx extension we can fingerprint this resource to .Net. Furthermore, the following two requests can help in identifying the underlying technology in a better manner on .Net.

```
HEAD / HTTP/1.0

HTTP/1.1 200 OK
Server: Microsoft-IIS/5.0
Date: Wed, 13 Oct 2004 18:28:45 GMT
X-Powered-By: ASP.NET
Connection: Keep-Alive
Content-Length: 7565
Content-Type: text/html
Set-Cookie: ASPSESSIONIDASSBTQAC=LIBHCGLCDKNLLKECPNLACMMB; path=/
Cache-control: private
```

The above request identifies servers running ASP.NET. The same request sent to web services resource (asmx) elicits this information.

```
HEAD /ws/customer.asmx HTTP/1.0

HTTP/1.1 500 Internal Server Error
Server: Microsoft-IIS/5.0
Date: Wed, 13 Oct 2004 18:29:07 GMT
X-Powered-By: ASP.NET
X-AspNet-Version: 1.1.4322
Cache-Control: private
Content-Type: text/html; charset=utf-8
```

Content-Length: 3026

As you can see we get added directive in the response "**X-AspNet-Version: 1.1.4322" .** This clearly specifies its version and we can say that the request gets served by an internal web service engine.

Similarly, Java Web Services runs with jws extension on a few platforms. By looking at this extension we can guess about the underlying backend technologies. Axis integrated with tomcat can be identified because of the jws extension.

*wsdl extension and query string*

WSDL(web services definition language) is the file in which web services' access information resides. To access web services, it is important to get a hold of this wsdl file and then generate a proxy for the same. WSDL enumeration is outside the scope of this paper. However this should get you started: a URL can have wsdl extension as a file extension or can be part of a querystring. Examples underlining this fact are listed below.

Example:
http://example.com/servlet/customer.access.wsdl
http://example.com/customer.asmx?wsdl
http://example.com/customer.asmx/wsdl

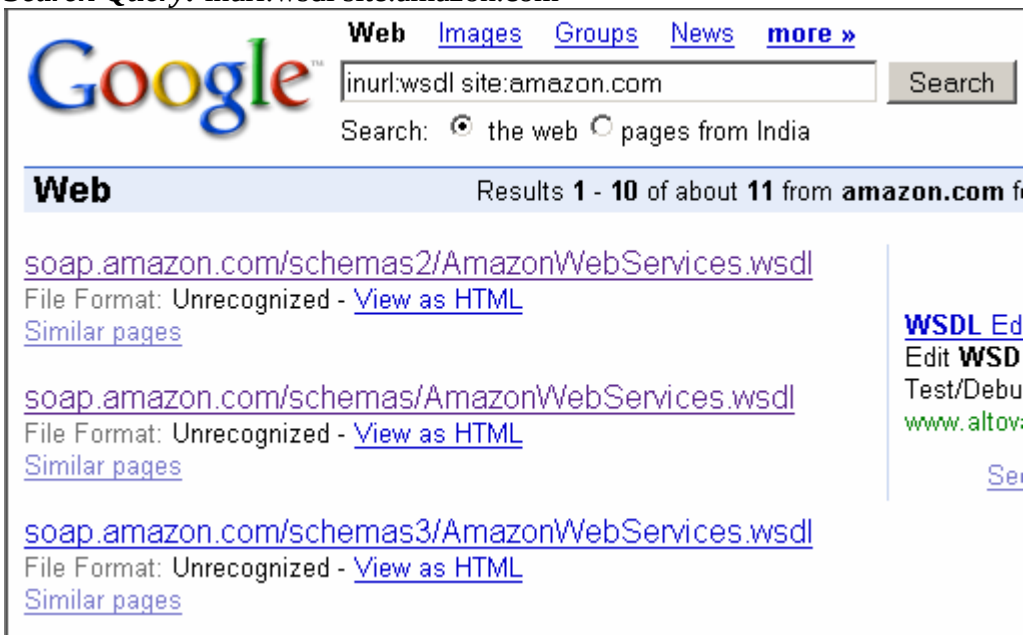
**5.3 Technology fingerprinting and discovery**

We have just seen how various extensions like asmx, jws and wsdl can be used to identify applications or sites running web services. We can crawl an entire application over HTTP and check for these extensions and profile web services. At the same time we can query the search engine database and try to locate web services. An example is illustrated below.

**Example:**

***Search Engine: Google.com***

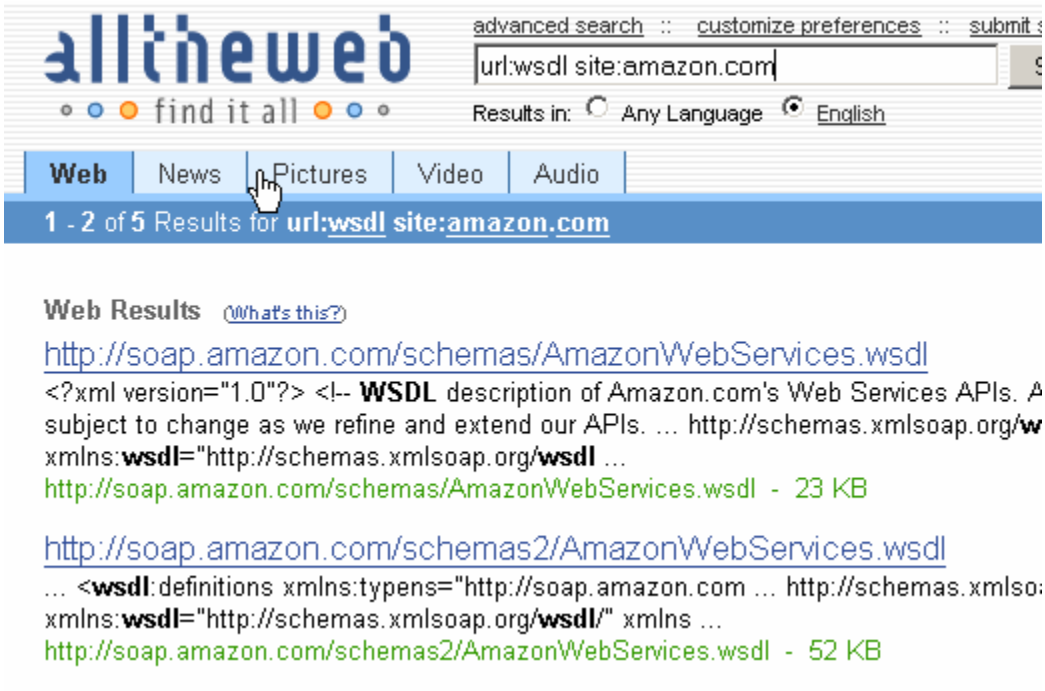
***Search Query:*** inurl:wsdl site:amazon.com




From above results we can locate sites running web services. Similarly, we can generate the following queries on Google.

1. filetype:wsdl site:amazon.com
2. inurl:asmx site:amazon.com
3. inurl:jws site:amazon.com

***Search Engine: alltheweb.com***
***Search Query:*** url:wsdl site:amazon.com



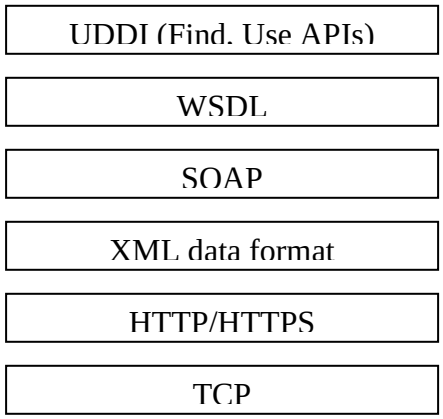

From the above results, we can obtain the list of sites offering web services.

# 6. Conclusion

The methods discussed in this paper will help you get started on Web service information gathering. There can be many more methods that can be identified and researched. UDDI and UBR can be the next generation “whois” server. The importance of UDDI, SOAP and WSDL is increasing since it is the key to next generation integration.

Footprinting, Discovery and Technology fingerprinting are the steps to gathering information about Web services offered by businesses. Enumeration using WSDL and attack vectors depending on information enumeration are the next two steps, both of which will be covered in separate papers and published at a later date. To complete an entire assessment phase of web services deployed at business locations, these three steps are all that are required to be understood and implemented.

**UDDI stack view**




**Inquiry APIs**

1. find_binding
2. find_business
3. find_relatedbusiness
4. find_service
5. find_tModel
6. get_bindingDetail
7. get_businessDetail
8. get_businessDetailExt
9. get_serviceDetail
10. get_tModelDetail

http://www.uddi.org

# References

[1] UDDI specifications – http://www.uddi.org
[2] Microsoft .Net XML Web Services (Microsoft .net) By Adam Freeman and Allen Jones
[3] SOAP specifications and RFCs - http://www.w3.org/TR/soap/
[4] WSDL Specifications and RFCs - http://www.w3.org/TR/wsdl