حملات روی Web Service ها، جمع آوری اطلاعات و کاوش روی Web Application ها

یکی از اولین گام ها در حملات علیه سیستم های اینترنتی و کامپیوتری، شناسایی هدف به طور کامل است. لذا بر اساس نیاز و درخواست بعضی دوستان بر آن شدم تا مقاله ای حول این مطلب بنویسم. مقاله ای که پیش رو دارید، مقاله ای صد در صد عملی و تست شده، راجع به آزمون ها و روش های عملی در شناسایی و جمع آوری اطلاعات راجع به Web Application ها و نیز حملات روی Web Service ها می باشد. در این روش می توانیم علاوه بر روش های معمول در شناسائی هدف، این روش ها را نیز امتحان کنیم.

لازم به ذکر است که این مقاله از دو قسمت اصلی تشکیل شده است. اولین قسمت آن، راجع به حملات روی وب سرویس ها است که حاوی بعضی روش های جمع آوری اطلاعات از جمله Footprinting و Discovery و Fingerprinting می باشد. قسمت دوم آن نیز راجع به footprint کردن و discovery روی Web Application ها بوده که در صورتی که Web Server ها، میزبانی چندین Web Application هستند، می توانید از این روش استفاده کنید.

قسمت اول: حملات روی Web Service ها

چکیده

سرویس های وب (Web Service)، با سرعتی بسیار بالا در حال رشد می باشند و به طبع مشکلات امنیتی جدیدی را در حیطه ی امنیت وب به ارمغان می آوردند. چطور شروع به تشخیص وب سرویس هایی کنیم که در هر شرکت گسترش یافته اند؟ این سوالی بنیادی است و تمامی این قبیل سوال ها و موضوعات، با جمع آوری اطلاعات[1] شروع خواهند شد. UDDI، WSDL و SOAP، سه رکن اساسی در این تکنولوژی هستند و ابزاری مفید برای جمع آوری اطلاعات می باشند. Universal Business Registry (UBR)[2]، با استفاده از UDDI به عمل Footprinting کمک می کند. UBR و Fingerprinting می توانند برای انجام کاوش در Web Service ها استفاده شوند. هدف این مقاله، تنها به اولین مرحله، یعنی مرحله جمع آوری اطلاعاتِ وب سرویس ها[3] محدود گشته است. در این مقاله، روش شناسی جمع آوری اطلاعات وب سرویس، بررسی شده است. دو مرحله بعدی ارزیابیِ روش شناسیِ جمع آوری اطلاعات، عبارتند از Enumeration و Defining Attack Vector که هر دوی آنها موضوعات گسترده ای می باشند که در مقاله های بعدی ادامه داده خواهند شد.

۱. مقدمه

۱/۱. زمینه

رشدِ تکنولوژی وب سرویس ها در پنج سال اخیر پدیده ای قابل ملاحظه بوده است. تکنولوژی های بسیاری در حال شکوفایی هستند، که از وب سرویس ها پشتیبانی می کنند و نیرویی برای اقتباس این گونه تکنولوژی ها را در بازار فراهم می سازد. Gartner و دیگر گروه های تحقیقاتی به شرکت ها توصیه می کنند که این تکنیک ها را بپذیرند و یا از عرصه رقابت بیرون رانده خواهند شد. مطالعات اخیر دلالت بر آن دارد که عرضه داشتِ وب سرویس[4]، در حال رشد بسیار سریعی می باشد به طوریکه از مقدار کنونی ۱/۶ میلیارد دلار، در سال ۲۰۰۷ به مقدار ۳۴ میلیارد دلار خواهد رسید.

افزایشِ رشدِ تکنولوژیِ وب سرویس ها و پذیرش هرچه سریع تر توسط شرکت هایی که به امید موفقیت های حرفه ای بیشتر فعالیت می کنند، مسائل امنیتی را در مرکز توجه آنان قرار داده است. Toolkit ها، آسیب پذیری ها[5] و اکسپلویت های جدیدی در حال سربرآوردن هستند که بویژه وب سرویس ها را هدف قرار می دهند. روش های حمله ی قدیمی تر کاربردی ندارند زیرا در شبکه امنیتی کاربردی Web Application که جدیدا توسعه یافته اند، از روش Application Layer Content Filtering استفاده به عمل می آید. وب سرویس ها هنوز هم دچار اغتشاش (و در هم بر همی) هستند، به دلیل اینکه از پروتکل های زیادی استفاده کرده و طیف گسترده ای از آسیب پذیری ها و مشکلات امنیتی را دامن می زنند.

---

[1] Information Gathering

[2] سیستم ثبت اطلاعات جامع

[3] Web Services Information Gathering Phase

[4] Web Service Offerings

[5] Vulnerability

۱/۲. فضای مشکلات

یک شبکه (و چارچوب) امنیتیِ سنتی، اساسا دارای سه سطح است:

سیستم های عملیاتی، سرویس ها و سطح کاربری[۶]

وب سرویس ها یک سطح جدید در چارچوب اضافه می کنند که لایه ی تجاری کاربری[۷] می باشد. فایروال ها در امر بلاک کردنِ ترافیکِ هدایت شده به سطح سیستم عملیاتی و سطح سرویس ها (Services Level) موثر هستند، اما قادر به بلاک کردن حملات در سطح Web Application[۸] نیستند، چرا که ترافیک روی پورت های ۸۰ و ۴۴۳ قانونی است.

دیوار آتش در لایه ی application، لایه ای از فیلترینگِ محتویات را فراهم می سازد. که حملاتی مانند SQL Injection و Parameter Tampering را بلاک خواهد کرد. اگرچه استفاده از آنها همواره توام با محدودیت هایی است. حملات جدیدی که وب سرویس ها را مورد هدف قرار می دهند، بدون دفاع مناسب، قابل جلوگیری نیستند. به همین دلیل است که مسائل امنیتی مربوط به وب سرویس ها، توجه افراد را به خود معطوف داشته اند. بعضی از چالش های در حال رشد در امنیتِ وب سرویس ها عبارتند از:

۱. Footprinting Web Services
۲. Gathering Information for Web Services
۳. Identifying Technologies
۴. Discovering endpoints

۱/۳. روش (Approach):

ارزیابی وب سرویس ها در مراحل زیر انجام می شوند:

الف)- Web Services Information Gathering
ب)- Web Services Enumeration
ج)- Web Services attack vector & defense strategies

این مقاله همان طور که در قبل ذکر شد، اولین مورد الف را در میان موارد فوق توضیح خواهد داد و فهمِ عملیاتِ جمع آوریِ اطلاعات وب سرویس ها، روش های نمونه و سازماندهی اطلاعات را آسان خواهد کرد. این روش، در ارزیابی web application های مختلف در استفاده از وب سرویس ها، مفید خواهد بود. موارد ب و ج نیز در مقاله های بعدی توضیح داده خواهند شد.

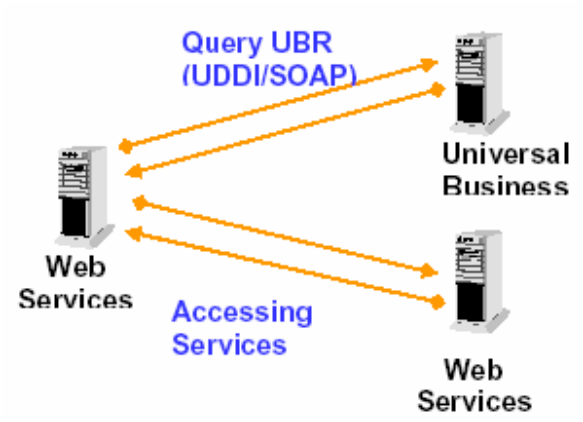
---

[6] Application level
[7] Business Application Layer
[8] Web Application Level Attacks

۲. متدلوژیِ[9] جمع آوریِ اطلاعاتِ وب سرویس

۲/۱. اصول پایه

وب سرویس ها، دارای سه سنگ بنای اساسی می باشند: UDDI، WSDL و SOAP. در این فرآیند دو بازیگر وجود دارد: مصرف کنندگان (استفاده کننده)[10] وب سرویس و تهیه کنندگان (ارائه دهنده)[11] وب سرویس. تمامی تعاملات بین این دو بازیگر، با استفاده از این سه سنگ بنا انجام می شوند. یک بازیگر نقش سومی وجود دارد که نقش حدواسط را بازی می کند و ارتباطات بین ارائه دهنده و مصرف کننده را فراهم می سازد که از آن با عنوان UBR – Universal Business Registry – یاد می کنیم.



عکس ۱: بازیگران و رفتارها

۲/۲. بازیگران، پروتکل ها و تعامل

همان طور که در شکل ۱ نشان داده شده، فرآیند زیر بین نهادهای مختلف رخ خواهد داد:

۱) استفاده کننده ی وب سرویس، از UBR گزارش گرفته و بسته به هر یک از نیازهایش بدنبال سرویس می گردد.

۲) UBR، لیست سرویس های در دسترس را ارائه می کند و استفاده کننده ی وب سرویس، یک یا چند تا از سرویس های *موجود را انتخاب می کند.

۳) استفاده کننده وب سرویس تقاضای یک نقطه دسترسی (Access Point) یا نقطه پایان (End Point) برای این سرویس ها خواهد داشت. UBR این اطلاعات را نیز عرضه می کند.

۴) استفاده کننده وب سرویس، آدرسِ IP یا آدرسِ Host مربوط به ارائه دهنده ی وب سرویس ها[12] را به دست آورده و دست یابی به سرویس را شروع خواهد کرد.

توضیحات زیر پروتکل های مورد استفاده در خلال هر مرحله از کل این فرآیند می باشد:

۱) UDDI (یا Universal Description, Discovery and Integration) – مراحل ۱ تا ۳ در این پروتکل رخ می دهند. سیستم پیغام رسانی SOAP، روی این پروتکل اجرا می شود.

---

[9] Methodology
[10] Consumer
[11] Supplier
[12] Web Services Supplier's Host/IP Address

۲) WSDL (Web Services Definition Language) – مرحله ۴ در فرآیند فوق، از این روش برای فهمِ چگونگیِ گسترش یافتنِ وب سرویس ها و نیز فهمِ اینکه چگونه این سرویس ها می توانند در سراسر شبکه در دسترس قرار گیرند استفاده می کند. لازم به ذکر است که WSDL در اینترنت، روی HTTP/HTTPS کار می کند.

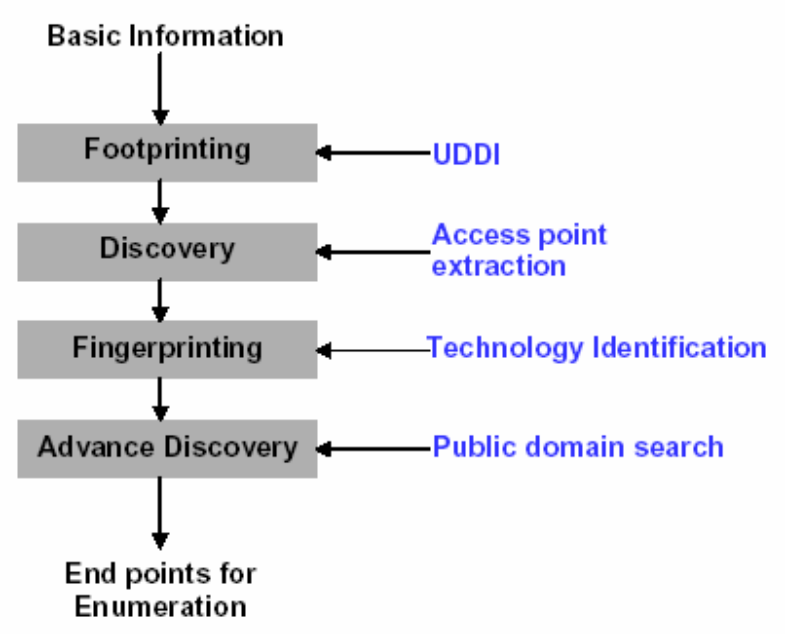
۳) SOAP (Simple Object Access Protocol) – این یک پروتکل ارتباطی است که بالاتر از پروتکل های HTTP/HTTPS و تعدادی از دیگر پروتکل ها، می نشیند. این پروتکل، در تمام فرآیند فوق، به عنوان یک سیستم برجسته در پیغام رسانی در ارتباطات[13] عمل می کند.

۲/۳. مراحل جمع آوری اطلاعات (Info. Gathering)

راه های گوناگونی برای جمع آوری اطلاعات در مورد یک سرویس با کمترین اطلاعات وجود دارد. این حداقل اطلاعات می تواند نام سازمان یا آدرس ایمیل یک کارمند باشد. در روش های سنتی، این جستجو برای اطلاعات با یک گزارش Whois انجام می شود که به دنبال آن تلاش برای به دست آوردن محدوده ی کل IP Address ها خواهد بود. در سال های اخیر، علاوه بر روش های سنتی، استفاده از روش های جدید در گزارش گیری از دیتابیسِ یک موتورِ جستجو باب شده است که اطلاعاتی بسیار حیاتی و عمیق به دست خواهند داد.

سوال کلیدی این است که: در یک وضعیت مشابه، چطور کار را وب سرویس ها آغاز کنیم؟

برای مثال، بیائید فرض کنیم که بیشترین اطلاعاتی که ما از هدف داریم، تنها، نام شرکت مورد نظر است. ما به دنبال اطلاعاتی از نوع وب سرویس این شرکت هستیم که آنها را برای بستن پیوندها و کلاینت ها ارائه می دهد. به یاد داشته باشید که به غیر از نام شرکت، ما هیچ اطلاعاتی اضافی در دسترس نداریم که بتوانیم با آنها کار را دنبال کنیم. اینجا روشی است که می توانیم برای جمع آوری اطلاعات مورد نیاز به کار گیریم.



عکس ۲: Methodology

[13] Underlying Communication Messaging System

همان طور که در شکل ۲ نشان داده شده است، جمع آوری اطلاعات می تواند با استفاده از مراحل زیر انجام شود:

۱. Web Services Footprinting

۲. Web Services Discovery

۳. Web Services Technology fingerprinting

۴. Advance discovery on public domain

مراحل لیست شده در فوق، هسته ی روشِ به دست آوردن اهدافمان و شناختن تهدیدهای وابسته با فاش شدن اطلاعات برای وب سرویس ها، را شکل دهی و قالب ریزی می کند. هر کدام از این موارد در عنوان های بعد در این مقاله، توضیح داده شده اند.

۳. عملیات Footprinting روی Web Service ها (Web Services Footprinting)

۳/۱. Footprinting

همان طور که در قسمت قبل بحث شد، Universal Business Registery (یا UBR)، یک منبع عمده از اطلاعات برای وب سرویس ها می باشد و جائی است که تاجرها (Business ها – یا موسسات تجارتی) وب سرویس ها را، ثبت (register) کرده و استفاده کننده (Consumer) به دنبال سرویس ها می گردد – یک دفتر ثبت عمومی - و در عمل بسیار شبیه به یک سرورِ Whois می باشد. گزارش (query) ها بهUBR فرستاده شده و در نتیجه پاسخ ها با اطلاعاتِ مورد نیاز برگشت داده می شوند. UBR روی خصوصیات و مشخصات UDDI و SOAP اجرا می شود.

وب سرویس ها، می توانند به وسیله ی یکی از سبک های زیر، روی UBR خریداری شوند (واقعا باید گفت برای هیچ از عبارات زیر، نمی توان کلمه یا عبارتی جالب در زبان فارسی پیدا کرد):

۱. Business Entity

۲. Business Service

۳. Binding Template

۴. Technical Model (یا tModel)

UBR ها، به وسیله شرکت های مختلفی از جمله Microsoft، IBM، SAP و ... ساخته می شوند. این شرکتها، اطلاعاتشان را با (در) یکی دیگر، کپی می کنند. مجموعه ای از API ها وجود دارد که می تواند در گزارش گیری از این registry، یاری رسان باشد. ما می توانیم از هر کدام از این API ها، برای گزارش گیری از UBR در مورد اطلاعات مخصوصی، استفاده کنیم. برای مثال، اگر به دنبال اطلاعاتی روی "amazon" هستیم، می توانیم شرکت را به وسیله همه ی این API ها، footprint کرده و ببینیم که چه اطلاعاتی می تواند استخراج شود. این API ها روی HTTP/HTTPS با (with – نه و)SOAP، کار می کنند. ویژگی های WSDL قبلا منتشر شده اند و می توانند به وسیله ی API های پرسشی (Inquire API)، به منظور استخراج اطلاعات ضروری استفاده شوند.

مجموعه های مختلفی از Toolkit ها و SDK ها، به وسیله شرکت های مختلفی از جمله Microsoft، Sun و ... منتشر و رشد یافته اند. این Toolkit ها، می توانند برای نوشتن برنامه هایی به منظورِ انجامِ عملیاتِ enumeration یا فقط درخواستِ API ها با استفاده از SOAP، روی کلاینت های ساده یTCP از جمله netcat، استفاده شوند. در زیر روشی ساده، برای تولیدِ یک درخواست با استفاده از یک کلاینتِ ساده ی TCP (TCP Client)، و فرستادن این درخواست روی شبکه، توضیح می دهیم. در زیر، http://uddi.microsoft.com ، درحقیقت، UBRِ ما برای footprinting کردنِ اطلاعات می باشد.

۳/۲. عملیات Footprinting روی Business Name (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:uddiorg:
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:uddiorg:
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="enus">
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-a95aeeb17e5910db"
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>
```

با آنالیز و تجزیه و تحلیل جواب و واکنش فوق، مشاهده می کنیم که node ها به ما اطلاعاتی روی هر business رجیستر شده با یک business name که حاوی رشته ی amazon باشد، می دهد. همچنین از بلاک اطلاعاتی فوق، ما می توانیم سرویس هایی که با business، وابسته هستند (و به پیوند داده شده اند) را تشخیص دهیم. دو business name زیر و registry key های مشابه با business در زیر یافت شده اند:

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

۳/۳. عملیات Footprinting روی Service Name (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:uddiorg:
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="enus">
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-448ab759-
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>
```

در پاسخ فوق، node های زیر به ما اطلاعاتی روی هر یک از سرویس های رجیستر شده با یک service name که حاوی رشته ی amazon باشد، می دهد. از بلاک اطلاعات فوق، می توانیم سرویس ها را تشخیص دهیم. Service name ها و service keys های متناظر و متشابه زیر، پیدا شده اند:

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>

۳/۴. عملیات Footprinting روی tModel (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:uddiorg:
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>
```

در جواب فوق، node های زیر اطلاعاتی روی هر tModel رجیستر شده با tModel Name که حاوی رشته ی amazon باشد، به ما می دهند. از بلاک اطلاعات فوق، می توانیم tModel ها را تشخیص دهیم. tModel و Key های مشابه و متناظر tModel در زیر یافت شده اند:

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

۳/۵. کاراکترها و ابزار Meta (Meta characters and Tools)

با روش های فوق می توانیم ابزاری ایجاد کنیم که به وسیله آن کار گزارش گیری از بیشتر از یک، **registry node** را روی اینترنت انجام داده که این کار، برای به دست آوردن تمامی اطلاعات ضروری، انجام می شود.این اطلاعات ممکن است مخصوص **business**، **service** و **tModel Name** ها باشد. برخلاف نام های ارائه دهنده[۱۴]، **UDDI** همچنین از بعضی از **meta character** ها نیز پشتیبانی می کند. برای مثال، علامت ٪ یعنی "شروع از" یا "Start From" که به این معنی است که اگر ما در حال جستجو روی نام هایی هستیم که با کلمه ی **amazon** شروع می شوند، می توانیم از "amazon%" به عنوان بخشی از گزارشمان استفاده کنیم.

[14] **Supplying Name**

۴. کاوش وب سرویس ها (Web Services Discovery)

۴/۱. کاوش (Discovery)

فرآیند **footprinting**، لیست سرویس های **register** شده روی **UBR** را برای دسترسی ما امکان پذیر می سازد. هدف کاوش، در حقیقت، شناختنِ نقاط دسترسی، برای هر کدام از این سرویس ها می باشد. یک نقطه ی دسترسی، حاوی یک **Host/IP Address** برای سرویس ها و مکان اصلی روی سرور می باشد. نقطه دسترسی به عنوان یک جزء کلیدی (**key component**) برای جمع آوری اطلاعات و **enumeration** کردن اطلاعات وب سرویس ها از ابتدا تا اکنون، نقش بازی می کند. مجددا باید گفت که همچنین **UBR** منبعی برای اطلاعات می باشد. کاوش، با استفاده از پروتکل یکسانی می تواند انجام شود و **UDDI** و **API** ها می توانند برای استخراج آن اطلاعات به کار گرفته شوند. در مثالمان نیز، در خلال مرحله ی **footprinting** ، **registry key** هایی برای **business**، **service** و **tModel** به دست آوردیم. این **key** می تواند برای تشخیص نقطه ی دسترسی وابسته با آن، استفاده شود.

۴/۲. کاوش روی Business Name (Discovery on Business Name)

برای انجام یک کاوش مبتنی بر business name، ابتدا باید سرویس های آن business name را بشناسیم. برای مثال، بیائید فرض کنیم که به دنبال سرویس هایی بر اساس business name می گردیم.

Amazon.com <18b7fde2-d15c-437c-8877-ebec8216d0f5>

در گام اول، ما سرویس های این **business name** را تعیین هویت می کنیم.

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:uddiorg:
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:uddiorg:
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>
```

۴/۳. کاوش روی سرویس ها (Discovery on services)

API: get_serviceDetail

در قسمت **footprinting** مشاهده کردید که نام سرویس (**service name**) را دریافت کردیم:

Amazon.com Web Services <ba6d9d56-ea3f-4263-a95a-eeb17e5910db>

همچنین در قسمت قبلی، توضیح داده شد که چطور همان **service key**، می تواند تنها با استفاده از **business name** دریافت شود. اکنون، با استفاده از این **service key** می خواهیم جزئیات سرویس را برداشت کنیم.

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-a95aeeb17e5910db</
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:uddiorg:
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</a
ccessPoint><tModelInstanceDetails
/></bindingTemplate></bindingTemplates></businessService></serviceDetail></
soap:Body></soap:Envelope>
```

در جواب فوق، URL این سرویس را به عنوان نقطه ی دسترسی خود، تعیین محل کرده ایم.

Discovery URL : http://soap.amazon.com/schemas/AmazonWebServices.wsdl

۴/۴. کاوش روی tModel (Discovery on tModel)

API: get_tModelDetail

در قسمت footprinting، ما tModel را دریافت کردیم:

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

اکنون با استفاده از این tModel Key می توانیم درخواست زیر را ارسال کنیم:

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>
```

در جواب فوق، یک URL را برای این tModel، به عنوان نقطه ی دسترسی خود، تعیین محل کرده ایم.

Discovery URL : http://soap.amazon.com/schemas2/AmazonWebServices.wsdl

۵. fingerprinting کردن فناوری وب سرویس ها (Web Services Technology Fingerprinting)

۵/۱. تکنولوژی fingerprinting و identification

یکی از چالش ها در دنیای امنیت، fingerprint کردن تکنولوژی ها و جمع آوری اطلاعات روی هر یک از این تکنولوژی ها است. و می تواند یک فرآیند در حال پیشرفت (و البته مداوم) باشد که مستلزم استفاده یا توسعه روش های زیادی است. در این جا، می خواهیم، به دنبال راه حلی برای این سوال باشیم: بعد از به دست آوردن یک URL کشف شده (Discovery URL)، تنها با نگاه کردن به رشته ی کاراکترها چه چیزی را می توانیم تشخیص دهیم؟ ما دو تکنولوژی را برای وب سرویس ها بررسی می کنیم: **.Net** و وب سرویس های جاوا که روی Axis اجرا می شوند[۱۵].

۵/۲. fingerprint کردن تکنولوژی با استفاده از پسوند ها (extension)

به عنوان یک مثال، بیائید دو URL زیر را به عنوان دو URL کشف شده (Discovery URL) در نظر بگیریم:

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

asmx/jws extension

این بخش از چارچوب NET/J2EE. منبعی برای وب سرویس ها بوده و وب سرویس ها می توانند با استفاده از این منبع رشد و گسترش داده شوند. از این رو، تنها با دستکاری اجمالی بر مجموعه از کاراکترها که حاوی پسوند asmx. هستند، می توانیم این منبع را به NET.، fingerprint کنیم. بعلاوه، دو درخواست زیر می توانند این تکنولوژی برجسته را در وضعیتی بهتر روی Net. مشخص کنند:

```
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
```

درخواست فوق نشان می دهد که سرورها روی ASP.NET در حال اجرا هستند. همان درخواست به منبع وب سرویس (asmx) فرستاده شد و نتایج زیر دریافت شدند:

```
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
```

[15] **Java Web Services running on Axis**

همان طور که می بینید ما Added Directory را در پاسخ گرفتیم: "X-AspNet-Version: 1.1.4322" که به وضوح نسخه اش را مشخص می کند و می توان گفت که درخواست به وسیله یک موتور داخلی وب سرویس[۱۶] به کار گرفته شده است.

به همین نحو، وب سرویس های جاوا (Java Web Services) با پسوند jws روی تعدادی پلاتفرم اجرا می شوند. با نگاه کردن به این پسوند می توانیم راجع به تکنیک های برجسته اش حدس بزنیم. Axis اگر با Tomcat یکپارچه شده باشد، به دلیل پسوند jws می تواند، تشخیص داده شود.

wsdl extension and query string

WSDL[۱۷]، فایلی است که در اطلاعات دستیابی وب سرور مستقر می باشد. برای دستیابی به وب سرویسها، خیلی مهم است که این فایلِ wsdl را گیر آورده و سپس یک پراکسی به همان نحو ایجاد کرد. WSDL Enumeration خارج از هدف این مقاله است.به هر حال، شروع کار به این صورت است:

یک URL می تواند پسوند wsdl مانند یک پسوند فایل داشته باشد یا می تواند بخشی از یک رشته ی گزارش (query string) باشد. مثال های برجسته در زیر لیست شده اند:

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

۵/۳. fingerprint و کاوش تکنولوژی

ما تا به حال تنها فهمیده ایم که چطور **extension** های مختلف مثل **asmx**، **jws** و **wsdl** می توانند برای تشخیص **application** ها یا سایت های در حال اجرای وب سرویس ها، استفاده شوند. می توانیم روی **HTTP** کل **application** را برای این موارد چک کرده و به دنبال این **extension** ها و **profile**ِ وب سرویس ها بگردیم. در همان زمان، می توانیم از دیتابیس یک موتور جستجو گزارش گرفته و سعی در تعیین محل کردن وب سرویس ها کنیم. یک مثال در زیر توضیح داده شده است.

مثال:

Search Engine: Google.com
Search Query: inurl:wsdl site:amazon.com

[16] **Internal Web Service Engine**
[17] **Web Services Definition Language**

از نتایج بالا، می توانیم سایت های در حال اجرای وب سرویس ها را تعیین محل کنیم. به همین نحو، مـی تـوانیم گـزارش هـای زیر را در گوگل تولید کنیم:

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

از نتایج فوق، می توانیم لیستی از سایت ها که وب سرویس ها را ارائه می دهند، به دست بیاوریم.

قسمت دوم: footprint و discovery روی Web Application

چکیده

تشخیص **Web Application** با آدرس **IP** و پورت ها (۸۰ / ۴۴۳) شروع می شود که عملی معمول بوده ولی در این روش یک نقص وجود دارد. اگر یک وب سرور با چندین هاست مجازی در حال اجرا باشد، تکلیف چه خواهد بود؟ به بیان دیگر، اگر یک سرور در حال اجرای بیشتر از یک **Web Application** باشد، چه باید کرد؟

در یک چنین سناریویی، تشخیص یک Web Application با استفاده از این قبیل ترکیب های IP/Port ممکن است به شکست انجامد و نتایج جزئی ارائه دهد. انجام یک Reverse DNS روی IP و استفاده از آن به عنوان HOST field در HTTP یک آزادی است، اما بیشتر موارد، به شکست می انجامد.

بنابراین، راه حل این مشکل چیست؟ راه حل با WHOIS Information Database و DNS Server شکل خواهد گرفت. این قسمت از مقاله، چگونگی به دست آوردن این اطلاعات را توضیح داده و فرآیند کاوش را برای Web Application ها پیگیری خواهد کرد.

حیطه مشکل

زمینه

بیائید مثالی را فرض کنیم که یک Web Server با آدرس IP برابر 203.88.128.10 (این آدرس فرضی است) روی اینترنت در حال اجرا است. پورت ۸۰ باز بوده و HTTP Traffic در حال ورود و خروج روی سرور است. این وب سرور ممکن است بیشتر از یک Web Application را میزبانی (hosting) کند (که به اصطلاح می گوئیم Hosting کند). یک مشخصه از پروتکل HTTP این است که HOST Information با هر درخواست از سرور خوانده شده و جواب دریافت شده از سرور، بستگی بر HOST Tag تهیه شده در HTTP Request دارد. برای مثال، بیائید فرض کنیم که یک وب سرور Apache روی 203.88.128.10 در حال اجراست. در این صورت httpd.conf آن شبیه به زیر خواهد بود:

httpd.conf on 203.88.128.10

```
<VirtualHost *:80>
    # ServerAdmin webmaster@dummy-host.example.com
    DocumentRoot /usr/local/apache2/htdocs
    # ErrorLog logs/dummy -host.example.com-error_log
    # CustomLog logs/dummy-host.example.com-access_log common
</VirtualHost>

<VirtualHost *:80>
    # ServerAdmin webmaster@dummy-host.example.com
    DocumentRoot /usr/local/apache2/htdocs/blue
    ServerName www.blue.com
    # ErrorLog logs/dummy -host.example.com-error_log
    # CustomLog logs/dummy-host.example.com-access_log common
</VirtualHost>

<VirtualHost *:80>
    # ServerAdmin webmaster@dummy-host.example.com
    DocumentRoot /usr/local/apache2/htdocs/red
    ServerName www.red.com
    # ErrorLog logs/dummy -host.example.com-error_log
    # CustomLog logs/dummy-host.example.com-access_log common
</VirtualHost>
```

از بلوک های فوق ما پی می بریم که Web Server در مدل هاست مجازی (Virtual Host Model) در حال اجراست. دو هاست تعریف شده اند و هر کدام، یک application در حال اجرا دارند:

www.red.com
www.blue.com

اولین بلوک از قسمت Virtual Host، *DocumentRoot* as */usr/local/apache2/htdocs,* را دارد که default root برای وب سرور می باشد – یک کلاینت فرستنده HTTP Request که بدون HOST Information است، به وسیله این root به کار گرفته می شود.

اکنون بیائید جواب های وب سرور را به سه درخواست مختلف از HTTP، ارزیابی کنیم.

Request:

```
C:\Documents and Settings\The Cephexin> nc 203.88.128.10 80
HEAD / HTTP/1.0
```

Response:

```
HTTP/1.1 200 OK
Date: Tue, 11 Jan 2005 20:17:40 GMT
Server: Apache/2.0.50 (Unix) mod_ssl/2.0.50 OpenSSL/0.9.7d mod_jk2/2.0.4
Content-Location: index.html.en
Vary: negotiate,accept-language,accept-charset
TCN: choice
Last-Modified: Fri, 04 May 2001 00:01:18 GMT
ETag: "1c4d0-5b0-40446f80;1c4e6-961-8562af00"
Accept-Ranges: bytes
Content-Length: 1456
Connection: close
Content-Type: text/html; charset=ISO-8859-1
Content-Language: en
Expires: Tue, 11 Jan 2005 20:17:40 GMT
```

درخواست فوق توسط default root بکار گرفته شد و ما یک default page با اندازه ی 1456 دریافت کردیم.

Request:

```
C:\Documents and Settings\The Cephexin> nc 203.88.128.10 80
HEAD / HTTP/1.0
Host: www.blue.com
```

Response:

```
HTTP/1.1 200 OK
Date: Tue, 11 Jan 2005 20:17:45 GMT
Server: Apache/2.0.50 (Unix) mod_ssl/2.0.50 OpenSSL/0.9.7d mod_jk2/2.0.4
Last-Modified: Tue, 04 Jan 2005 23:10:29 GMT
ETag: "1865-b-f991a340"
Accept-Ranges: bytes
Content-Length: 11
Connection: close
Content-Type: text/html; charset=ISO-8859-1
```

درخواست فوق توسط *DocumentRoot /usr/local/apache2/htdocs/blue directive* به کار گرفته شد. اندازه صفحه ای برابر با ۱۱ به web application وابسته به هاستِ www.blue.com فرستاده شد. این نشان می دهد که یک applicationِ کاملا جدید، توسط وب سرور که بستگی به Host Tag در درخواست GET/HEAD/POST دارد، به کار گرفته شده است. که در این مورد، درخواست، HEAD بود.

Request:

```
C:\Documents and Settings\The Cephexin> nc 203.88.128.10 80
HEAD / HTTP/1.0
Host: www.red.com
```

Response:

```
HTTP/1.1 200 OK
Date: Tue, 11 Jan 2005 20:17:57 GMT
Server: Apache/2.0.50 (Unix) mod_ssl/2.0.50 OpenSSL/0.9.7d mod_jk2/2.0.4
Last-Modified: Tue, 04 Jan 2005 23:16:57 GMT
ETag: "1cc0b-9-10b20c40"
Accept-Ranges: bytes
Content-Length: 9
Connection: close
Content-Type: text/html; charset=ISO-8859-1
```

درخواست فوق توسط *DocumentRoot /usr/local/apache2/htdocs/red directive* به کار گرفته شده است. اندازه صفحه ای برابر با ۹ برای web application وابسته به هاستِ www.red.com فرستاده شد. یعنی یک applicationِ کاملا جدید، توسط وب سرور که بستگی به Host Tag در درخواست GET/HEAD/POST دارد، به کار گرفته شده است. که در این مورد، درخواست، HEAD بود.

مسئله

در مثال توضیح داده شده در فوق، ما فرض می کنیم که به فایل پیکربندی آپاچی[18] یعنی httpd.conf دستیابی داریم. بر اساس این فرض، می توانیم یک Tag "Host" را با مقدار فعلی بفرستیم. اما شاید در واقعیت این طور نباشد. تمام اطلاعاتی که در دسترس شما است، تنها یک آدرس IP (مثال: 203.88.128.11) یا نام هاست (مثال: www.yahoo.com) می باشد.

ما باید قادر به ایجاد لیستی از هاست هایی مقدوری که به همان آدرس IP اشاره دارند را با استفاده از این اطلاعات کم، باشیم. چطور می توانیم اطلاعاتی بیشتر جمع آوری کنیم؟ این مشکل در این مقاله توضیح داده شده است.

راه حل

گام اول: پیدا کردن Nameserver ها برای یک آدرس IP بخصوص

اولین هدف ما، پیدا کردن یک **nameserver** برای یک آدرس **IP** بخصوص است. برای مثال آدرس **203.88.128.10** را در نظر می گیریم. ما احتیاج به پیدا کردن بلوکی که این آدرس **IP** به آن ارجاع داده شده و نیز پیدا کردنِ **nameserver** برای این آدرس **IP** بخصوص داریم، به طوریکه می توانیم چندین گزارش را در صورت نیاز اجرا کنیم. اجرا کردن گزارش زیر روی یک دیتابیس **ARIN**، نتایج زیر را بدنبال داشت:

```
C:\Program Files\GnuWin32\bin>jwhois -h whois.arin.net 203.88.128.10
[Querying whois.arin.net]
[whois.arin.net]

OrgName: XYZ corp
OrgID: XYZC
Address: 101 First Avenue
City: NYC
StateProv: NY
PostalCode: 94089
```

[18] **Apache Configuration File**

```
Country: US

NetRange: 203.88.128.0 – 203.88.128.255
CIDR: 203.88.128.0/20
NetName: XYZC-4
NetHandle: NET-203-88-128-0-1
Parent: NET-203-0-0-0-0
NetType: Direct Allocation
NameServer: ns1.xyz.com
NameServer: ns2.xyz.com
Comment:
RegDate: 2003-07-17
Updated: 2003-07-17

OrgTechHandle: NA098-ARIN


OrgTechName: Netblock Admin
OrgTechPhone: +1-212-999-9999
OrgTechEmail: netblockadmin@xyz.com

# ARIN WHOIS database, last updated 2005-01-10 19:10
# Enter ? for additional hints on searching ARIN's WHOIS database.

C:\Program Files\GnuWin32\bin>
```

اکنون ما اطلاعات nameserver **را برای این آدرس** IP **داریم. ما می توانیم یک** port scan **را روی کل محدوده** (range) **انجام داده و همچنین به دنبال پورت** 53 UDP **بگردیم. این کار هر** nameserver **ممکن را که در حال اجرا روی این محدوده است به ما خواهد داد. اکنون ما یک یا چند آدرس** IP **داریم که مانند** nameserver **ها برای آدرسِ** IP **هدف ما در حال اجرا هستند. این** nameserver **ها می توانند برای موارد** enumeration **مورد استفاده قرار گیرند.**

گام دوم: جستجو به دنبال رکوردهای PTR روی nameserver

اکنون، ما از nslookup استفاده کره و سعی بر گزارش گیری از PTR Record ها برای آدرس IP هدف می کنیم. همان طور که در شکل زیر نشان داده شده است:

```
C:\Documents and Settings\The Cephexin>nslookup
Default Server: ns1.icenet.net
Address: 203.88.128.7

> server ns1.xyz.com
Default Server: [203.88.128.250]
Address: 203.88.128.250

> 203.88.128.10
Server: [203.88.128.250]
Address: 203.88.128.250
Name: www.blue.com
Address: 192.168.7.50

> set type=PTR
> 203.88.128.10
Server: [203.88.128.250]
Address: 203.88.128.250

10.128.88.203.in-addr.arpa name = www.blue.com
10.128.88.203.in-addr.arpa name = www.red.com
>
```

ما قادریم که PTR Record آنها را بدست آوریم. می توان دید که دو domain وجود دارند که به همان آدرس IP اشاره دارند. همین دستورالعمل می تواند به وسیله استفاده از دستور dig نیز تکرار شود.

گام سوم: PTR Record با شکست مواجه می شود

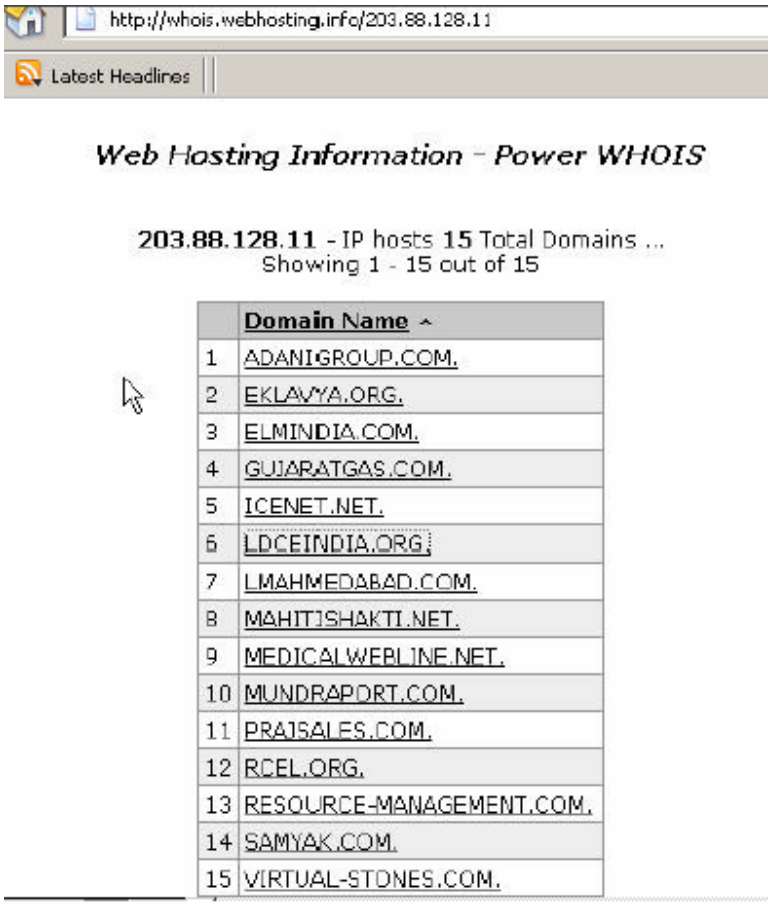
بسیاری اوقات نمی توانیم PTR Record های روی nameserver بدست آوریم یا entry ها، به سادگی روی nameserver ایجاد نشده اند. در این مورد، نمی توانیم اطلاعات صحیحی درباره آدرس IP هدف، تعیین کنیم. یکی از روش ها برای حل این مشکل پیچیده، گزارش گرفتن از یک WHOIS سرور می باشد.

یک گزارش عادی whois، توسط پروتکل whois پشتیبانی شده و به دلیل اینکه یک چنین گزارشی پشتیبانی نشده است، سرور برای دریافتِ نوعِ اطلاعات جستجو شده، در حجم و اندازه ی خودش محدود شده است. اما، database های تعدادی از whois server، این database ایجاد شده را دارند. سروکله زدن با این database ها برای footprint کردن web application بسیار ضروری است. این گزارش می تواند Reverse IP Query نامیده شود. یکی از DB هایی که می توانیم گزارش گیری کنیم، webhosting.info می باشد.

برای توضیح چگونگی انجام آن، بیائید از آدرس IP برابر با 203.88.128.11 (IP حقیقی روی اینترنت) استفاده کنیم که هیچ PTR Record روی nameserver ندارد.

```
C:\Documents and Settings\The Cephexin>nslookup
Default Server: ns1.icenet.net
Address: 203.88.128.7

> server 203.88.128.250
Default Server: icedns1.icenet.net
Address: 203.88.128.250

> 203.88.128.11
Server: icedns1.icenet.net
Address: 203.88.128.250
Name: ice.128.client11.icenet.net
Address: 203.88.128.11

> set type=PTR
> 203.88.128.11
Server: icedns1.icenet.net
Address: 203.88.128.250
Non-authoritative answer:
11.128.88.203.in-addr.arpa      name = ice.128.client11.icenet.net

> 203.88.128.11
Server: icedns1.icenet.net
Address: 203.88.128.250
Non-authoritative answer:
11.128.88.203.in-addr.arpa      name = ice.128.client11.icenet.net
>
```

هنوز تمام نشده است. هنوز ما شماره هاست های موجود روی سرور را به دست نیاوردیم. اما می توانیم یک گزارش whois روی webhosting.info داشته باشیم. به نتایج زیر نگاه کنید (این یک IP زنده (Live IP) برای یک ISP است).



نام پانزده هاست به همان آدرس IP اشاره دارد. اکنون می دانیم که پانزده application در حالِ اجرا روی این سرور هستند. این database توسط service provider در خلال یک دوره زمانی استنتاج شده و ممکن است درست نباشد، اما شروع مناسبی برای مرحله کاوش (discovery) است.

مرحله چهارم: کاوشِ هر یک از این هاست ها (Discoverying)

ما می توانیم یک **HEAD Request** به یک آدرس **IP**، با استفاده از یکی از هاست های بالا بفرستیم و ببینیم که چه جوابی از سرور می گیریم.

Request 1 [Default]:

```
C:\Documents and Settings\The Cephexin>nc 203.88.128.11 80
HEAD / HTTP/1.0
```

Response:

```
HTTP/1.1 404 Object Not Found
Server: Microsoft-IIS/4.0
Date: Thu, 27 Jan 2005 10:12:16 GMT
Content-Type: text/html
Content-Length: 102

<html><head><title>Error</title></head><body>The system cannot find the file specified. </body></html>
```

می توانیم یک درخواست را با هیچ HEAD تعیین شده ای بفرستیم و یک جوابِ 404 را دریافت کنیم.

Request 2 [*junk* as host]:

```
C:\Documents and Settings\The Cephexin>nc 203.88.128.11 80
HEAD / HTTP/1.0
Host: junk
```

Response:

```
HTTP/1.1 404 Object Not Found
Server: Microsoft-IIS/4.0
Date: Thu, 27 Jan 2005 10:14:37 GMT
Content-Type: text/html
Content-Length: 102

<html><head><title>Error</title></head><body>The system cannot find the file specified. </body></html>
```

در این درخواست، ما junk را به عنوان یک مقدار برای Host Tag فرستادیم و مجددا 404 را دریافت کردیم.

Request 3:

```
C:\Documents and Settings\The Cephexin>nc 203.88.128.11 80
HEAD / HTTP/1.0
Host: icenet.net
```

Response:

```
HTTP/1.1 200 OK
Server: Microsoft-IIS/4.0
Content-Location: http://icenet.net/index.htm
Date: Tue, 11 Jan 2005 10:07:12 GMT
Content-Type: text/html
Accept-Ranges: bytes
Last-Modified: Wed, 05 Jan 2005 06:52:02 GMT
ETag: "0553fff3f2c41:b3ae6"
Content-Length: 33442
```

در این مثال ما icenet.net را به عنوان Host Value فرستادیم که از مرحله سوم مشتق گرفته و یک جواب 200 دریافت کردیم که در آن یک content length برابر با 33442 و تعدادی ETag Value، دریافت کردیم.

Request 4:

```
C:\Documents and Settings\The Cephexin>nc 203.88.128.11 80
HEAD / HTTP/1.0
Host: adanigroup.com
```

Response:

```
HTTP/1.1 200 OK
Server: Microsoft-IIS/4.0
Content-Location: http://adanigroup.com/index.htm
Date: Tue, 11 Jan 2005 10:07:24 GMT
Content-Type: text/html
Accept-Ranges: bytes
Last-Modified: Wed, 28 Apr 2004 14:51:55 GMT
ETag: "80771d59302dc41:b3ae6"
Content-Length: 806
```

به همین نحو، می توانیم به adanigroup.com دستیابی پیدا کنیم.

Request 5:

```
C:\Documents and Settings\The Cephexin>nc 203.88.128.11 80
HEAD / HTTP/1.0
Host: www.mundraport.com
```

Response:

```
HTTP/1.1 200 OK
Server: Microsoft-IIS/4.0
Content-Location: http://www.mundraport.com/index.htm
Date: Tue, 11 Jan 2005 10:09:56 GMT
Content-Type: text/html
Accept-Ranges: bytes
Last-Modified: Thu, 01 Jul 2004 05:59:09 GMT
ETag: "80f45486305fc41:b3ae6"
Content-Length: 607
```

ما همچنین به www.mundraport.com دسترسی پیدا کردیم.

با ادامه این روش، می توانیم لیستی از همه هاست های ترسیم شده (mapped) به یک آدرس IP مشخص را روی سرور داشته باشیم. هر هاست می تواند با هدف ارزیابی، به عنوان یک web application مجزا تقی شود.

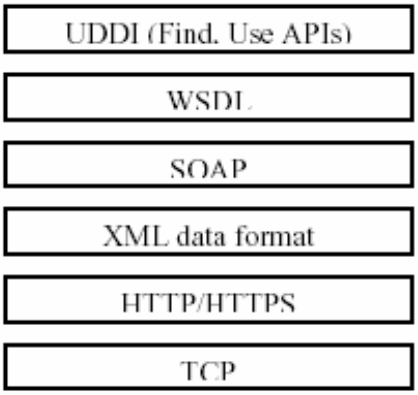
۶. نتیجه (Conclusion)

روش های بحث شده در این مقاله، به شما در راستای جمع آوری اطلاعات در مورد وب سرویس ها کمک خواهد کرد. هرچند روش های بیشتری برای این کار وجود دارند. UDDI و UBR می توانند، مولد، Whois Server های بعدی باشند. اهمیت UDDI، SOAP و WSDL در حال فزونی است.

Footprinting، Discovery و تکنولوژی fingerprinting، مراحل جمع آوری اطلاعات در مورد وب سرویس های ارائه شده توسط شرکت ها هستند. عملیات Enumeration با استفاده از WSDL و Attack Vector های مبتنی بر info. Enum دو گام بعدی هستند، که هر دو در مقالات بعد بحث خواهند شد. برای کامل کردن یک مرحله از تشخیص وب سرویس های گسترش یافته روی مکان های شرکتی، این سه مرحله، مراحلی لازم هستند که باید درک شده و به درستی اجرا شوند.

Appendix A: UDDI Quick Reference

UDDI stack view



API های گزارشی[19]:

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

منابع و ارجاع ها

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

نویسنده: سعید بیکی (cephexin@secumania.net)

© Secumania Security & Vulnerability Research Lab
www.secumania.net

[19] **Inquiry**