Oyedata User Guide

Oyedata for OData Assessments v1.0

Author:

Gursev Singh Kalra
Principal Consultant
Foundstone Professional Services

Table of Contents

Introduction

The Open Data Protocol (OData[1][2]) is an open web protocol for querying and updating data. OData enables the creation of HTTP based RESTful[3] data services that can be used to publish and edit resources identified using Uniform Resource Identifiers (URIs) with simple HTTP messages. OData is intended to be used to expose and access information from a variety of sources including, but not limited to, relational databases, file systems, content management systems, and traditional web sites. It allows a consumer to query a data source over HTTP protocol and get results back in formats like Atom, JSON or plain XML. OData can be termed as JDBC/ODBC for the internet.

The OData protocol does not include any security specifications but instead suggests that its implementers use what best fits their target scenario. As more applications, websites, and frameworks support OData, a larger attack surface becomes available to attackers. This paper discusses Foundstone's Oyedata – a new free tool to perform black-box OData security testing and help secure OData deployments.

Oyedata Features

Oyedata contains a number of features in order to facilitate OData security testing, the major features are summarized below:

1. Intuitive GUI based tool written in C#.
2. Ability to create attack templates from local and remote Service Documents and Service Metadata Documents.
3. Support for XML and JSON data formats.
4. Ability to export attack templates in JSON and XML formats that can be fed to custom Fuzzing code.
5. Ability to engage the OData services for manual testing.
6. Data generator for EDMSimpleType test data generation.
7. Ability to generate "Read URIs" for Entities, Entity Properties and Entity Property Values.
8. Ability to generate attack templates for Creation of new Entries, updating existing Entries, Service Operation invocation, Entry deletion etc…
9. Ability to identify Keys, Nullable and Non-Nullable Properties and indicate the same in the attack templates.

[1] http://www.odata.org/

[2] For more background on Odata, visit http://www.foundstone.com for the "Hacking OData" whitepaper.

[3] http://en.wikipedia.org/wiki/Representational_state_transfer

10. Web proxy, HTTP and HTTPS support.
11. Error logging.

Oyedata Installation

Oyedata has been tested on Windows 7 and XP. It requires the Microsoft .Net 4.0 Framework which can be downloaded from Microsoft's website[4]. Double clicking the setup.exe launches the self explanatory Oyedata setup.

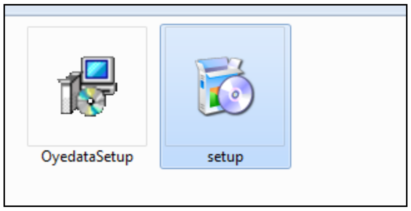
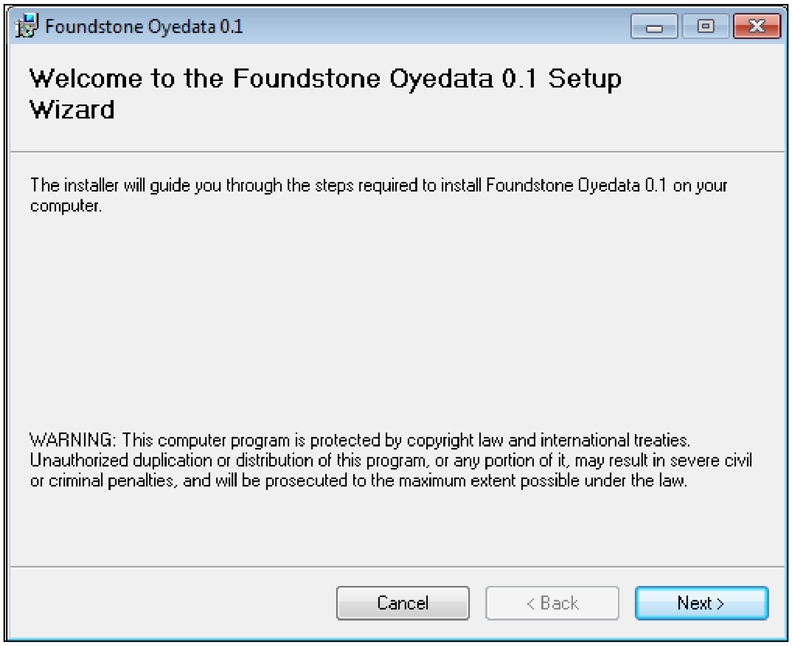


Figure 1: Image shows Oyedata Setup files



Welcome to the Foundstone Oyedata 0.1 Setup Wizard

The installer will guide you through the steps required to install Foundstone Oyedata 0.1 on your computer.

WARNING: This computer program is protected by copyright law and international treaties. Unauthorized duplication or distribution of this program, or any portion of it, may result in severe civil or criminal penalties, and will be prosecuted to the maximum extent possible under the law.

Figure 2: Image shows Oyedata setup screen

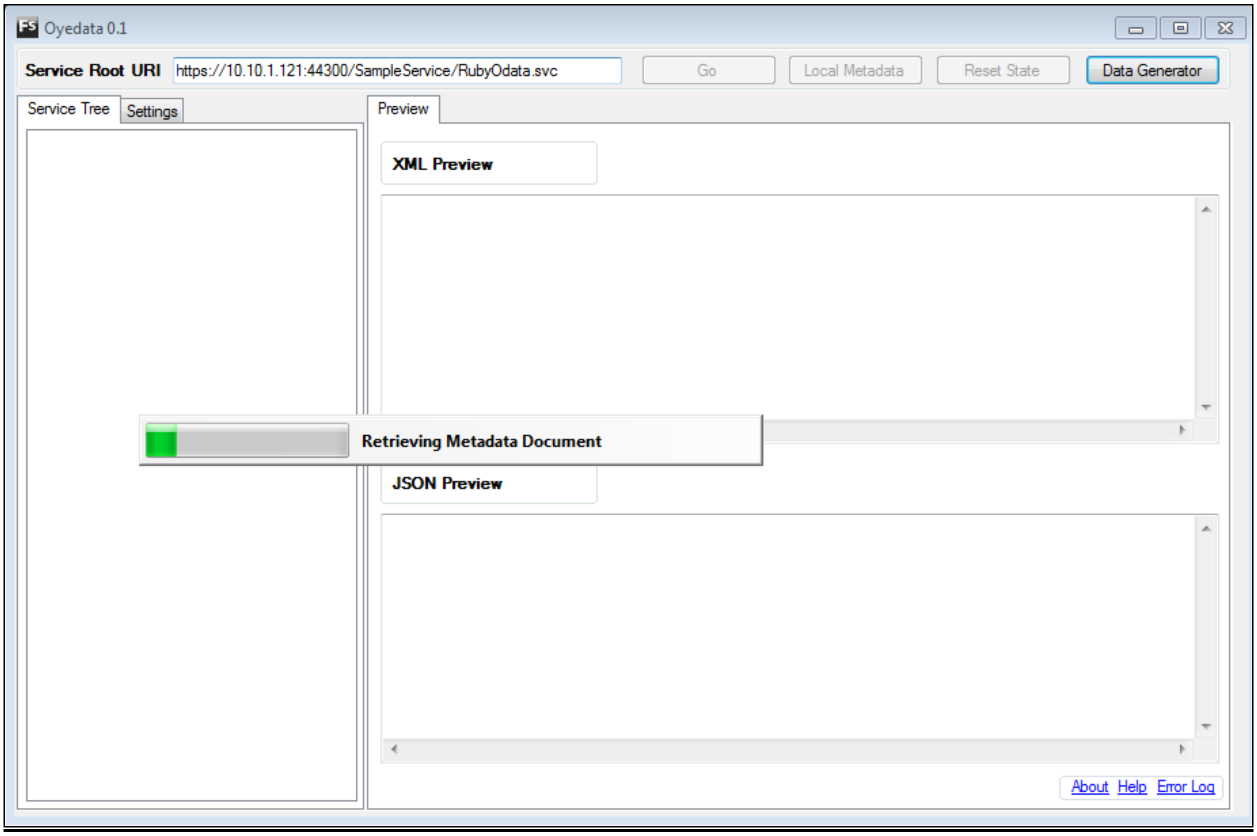
Bootstrapping Oyedata

The first step towards engaging an OData service is to analyze the OData Metadata document. During the analysis stage, Oyedata retrieves/loads and parses the metadata document to construct OData requests for CREATE, READ, UPDATE and DELETE operations.

[4] http://www.microsoft.com/en-us/download/details.aspx?id=17718

Remote Service Metadata Document

This is the default operation mode where we provide Oyedata with Service Root URI and hit the "Go" button. Oyedata retrieves the Service Metadata Document from the `/$metadata` relative path from the web server and analyzes it to construct the request tree.



Service Root URI https://10.10.1.121:44300/SampleService/RubyOdata.svc

Retrieving Metadata Document

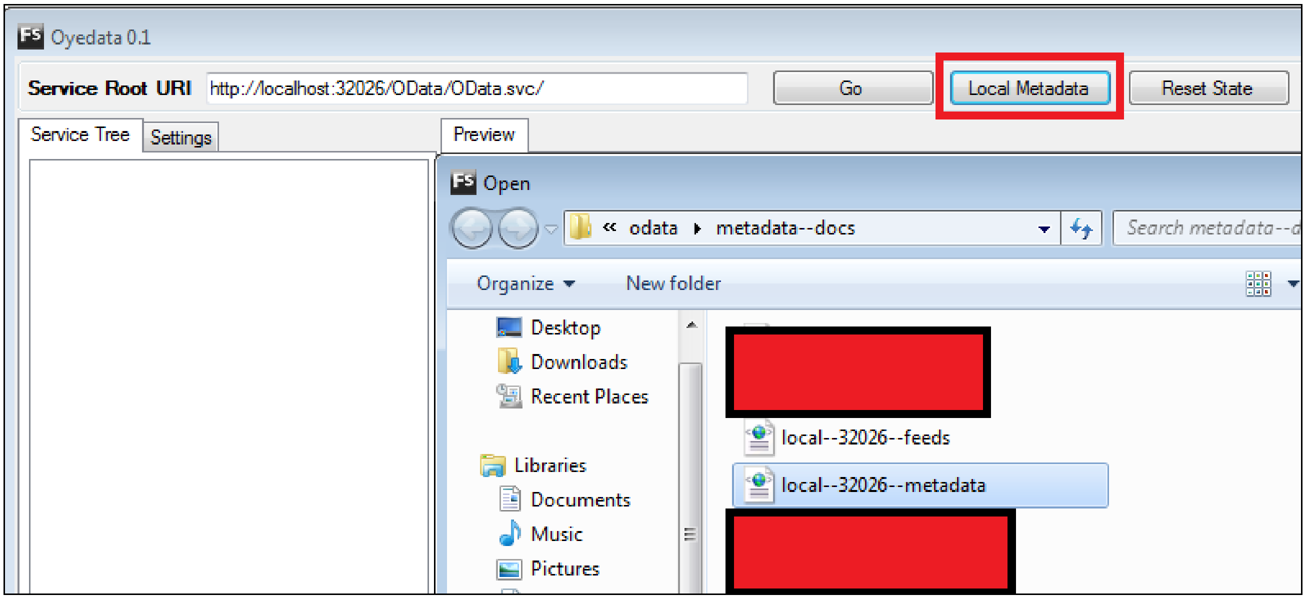
Figure 3: Image shows Oyedata retrieving an OData Service Metadata document

Local Service Metadata Document

The web server may be configured to restrict access to the service metadata document, in this case, Oyedata has the option to import it if it was obtained some other way (e.g. requested from the application owner). To manually import the service metadata document:

1. Click on the "Local Metadata" button, locate and select the metadata file.
2. Enter a Service Root URI and hit the "Go" button. It is important to point out that Service Root URI is mandatory for Oyedata to find the target OData service and it's an error to not provide a valid Service URI.

Upon hitting the "Go" button Oyedata loads the local metadata file and uses it along with the supplied Service Root URI to construct request templates.



Oyedata 0.1

Service Root URI http://localhost:32026/OData/OData.svc/

Figure 4: Image shows "Local Metadata" button to feed locally stored metadata document fed to Oyedata

Resetting Oyedata State

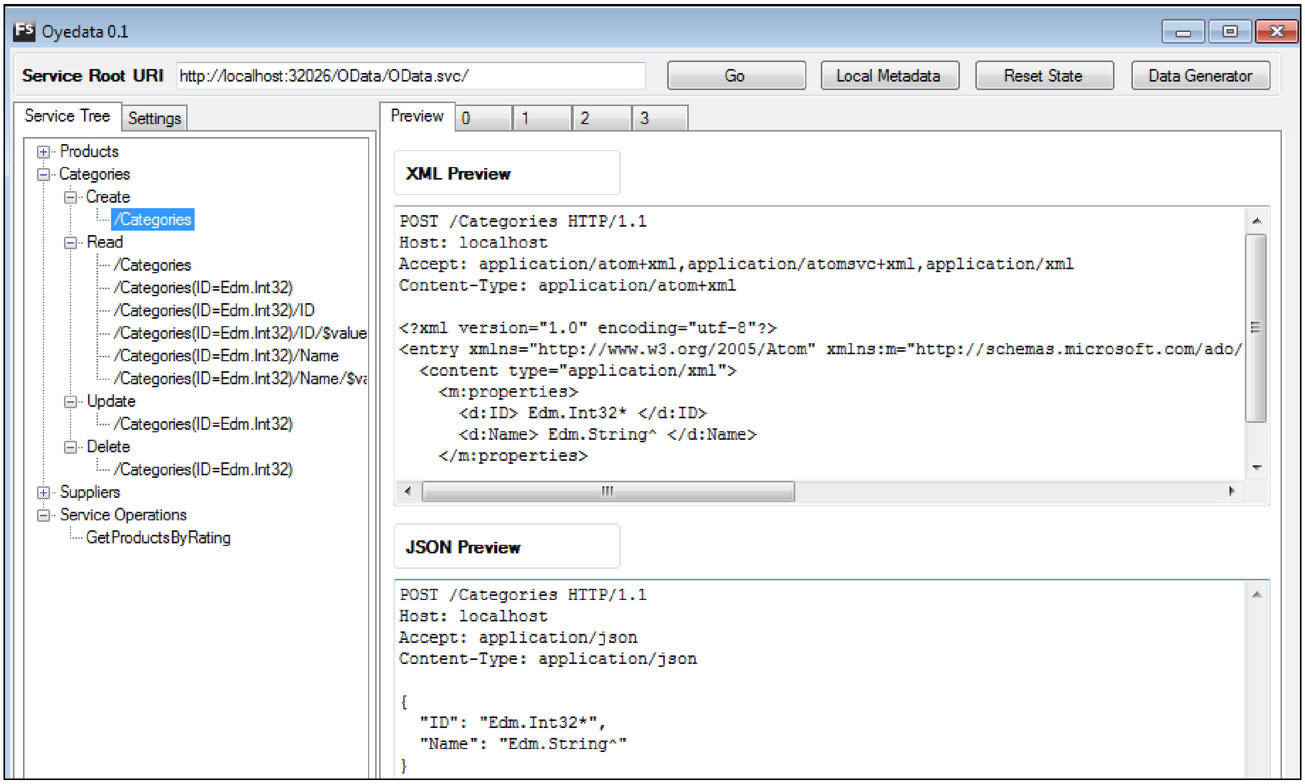
Clicking on the "Reset State" button clears the Oyedata state and performs the following actions:

1. Clears the in-memory metadata document (whether it is remotely retrieved or read from the local file system).
2. Removes the Request Tree
3. Deletes all the request tabs
4. Clears contents from JSON and XML preview boxes

Engaging an OData Service

Once metadata document analysis is complete and service request tree is created, the next step is to review the request templates, edit them, and start sending requests to the OData service. The service tree organization is explained below:

1. It contains one top level node for every Feed exposed by the OData service.
2. CREATE, READ, UPDATE and DELETE operation templates are available under each Feed for the Entity Type they expose.
3. All Service Operations are combined under a separate top level node.
4. Leaf nodes contain the service engagement templates.


Oyedata 0.1
Service Root URI
http://localhost:32026/OData/OData.svc/
Go
Local Metadata
Reset State
Data Generator
Service Tree
Settings
Products
Categories
Create
/Categories
Read
/Categories
/Categories(ID=Edm.Int32)
/Categories(ID=Edm.Int32)/ID
/Categories(ID=Edm.Int32)/ID/$value
/Categories(ID=Edm.Int32)/Name
/Categories(ID=Edm.Int32)/Name/$va
Update
/Categories(ID=Edm.Int32)
Delete
/Categories(ID=Edm.Int32)
Suppliers
Service Operations
GetProductsByRating
Preview
0
1
2
3
XML Preview
JSON Preview

```
POST /Categories HTTP/1.1
Host: localhost
Accept: application/atom+xml,application/atomsvc+xml,application/xml
Content-Type: application/atom+xml

<?xml version="1.0" encoding="utf-8"?>
<entry xmlns="http://www.w3.org/2005/Atom" xmlns:m="http://schemas.microsoft.com/ado/
  <content type="application/xml">
    <m:properties>
      <d:ID> Edm.Int32* </d:ID>
      <d:Name> Edm.String^ </d:Name>
    </m:properties>
```

```
POST /Categories HTTP/1.1
Host: localhost
Accept: application/json
Content-Type: application/json

{
  "ID": "Edm.Int32*",
  "Name": "Edm.String^"
}
```

Figure 5: Image shows various Create, Read, Delete and Update operations of the "Categories" Feed along with the supported Service Operation nodes

Reviewing Request Templates

Clicking on the leaf nodes populates the non-editable XML and JSON request preview areas with requests that will be sent to the OData service. The preview areas allow users to view the data types, parameter names, request headers, etc… that form the request. Additional parameter specific information is also presented:

1. If a parameter is an Entry Key, the data type is appended with an asterisk (*) symbol.
2. If a parameter is Nullable, the data type name is appended with a caret (^) symbol.

Generating Request Templates

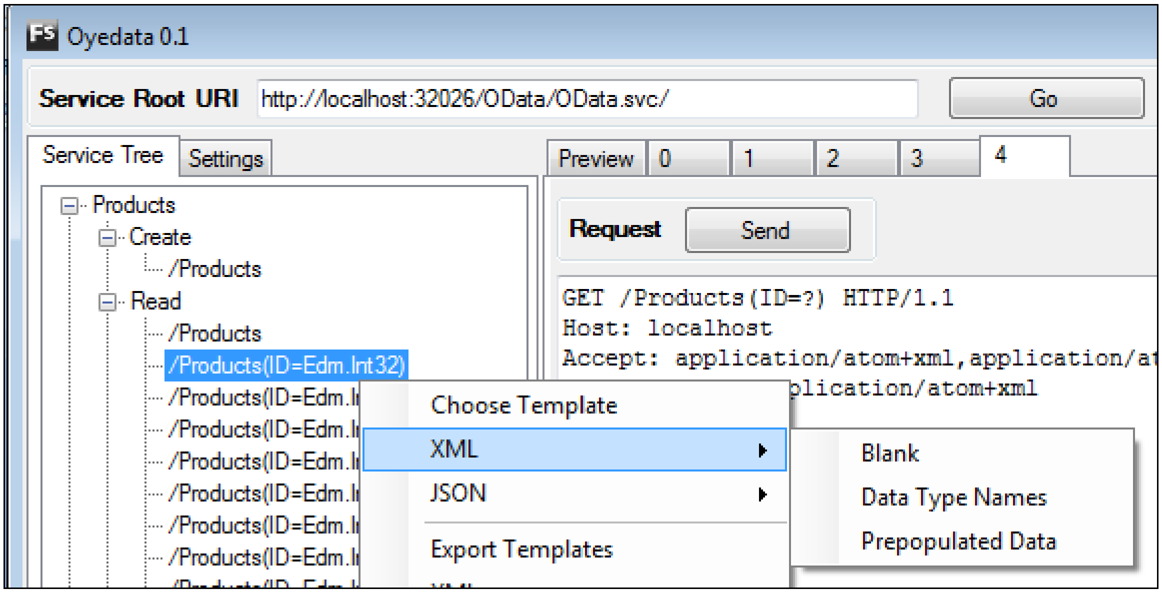
Once you have chosen the request you want to send to the OData service, the next step is to generate a request template. Oyedata provides options to generate three different types of request templates in either JSON or XML format. Right click on the leaf node, then under "Choose Template", then pick the template from JSON or XML sections. There are three template options:

1. **Blank:** All input fields as indicated by the service metadata documents are designated with question (?) symbols. Double clicking on a leaf node creates a new tab and populates it with a blank template.
2. **Data Type Names:** Information from the metadata document is used to pre-populate all input fields values with data types accepted by the fields like `Edm.Int`, `Edm.Guid`, `Edm.DateTime` etc… along

with Nullable and Key information. The generated templates can then be edited before engaging the service.

3. **Pre-Populated Data Types:** When this option is selected, pre-populated data of matching data types for each input field is populated in the generated templates. This may save you a lot typing time over an entire test.

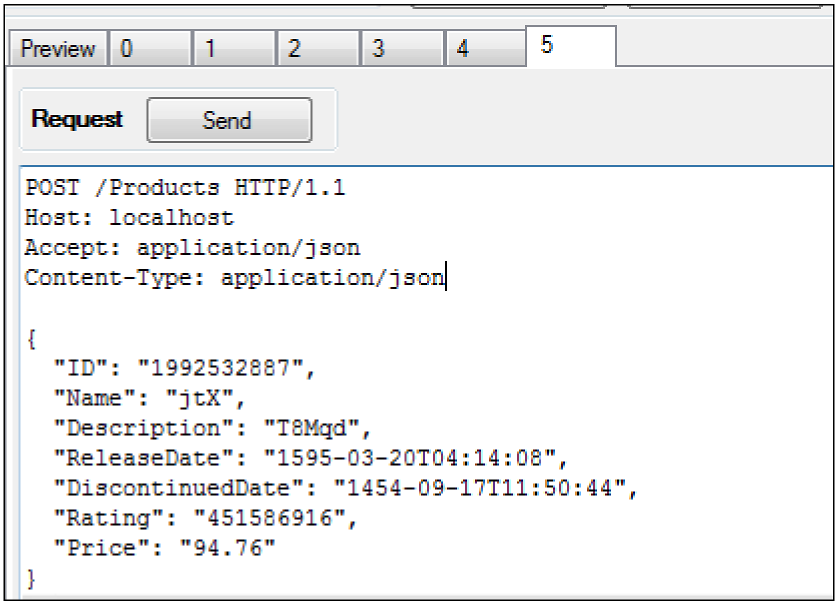
Once a template selection is made, Oyedata creates a new tab with editable request and response areas along with a Send button.



Service Root URI: http://localhost:32026/OData/OData.svc/

```
GET /Products(ID=?) HTTP/1.1
Host: localhost
Accept: application/atom+xml,application/at
                          plication/atom+xml
```

Figure 6: Image shows the three different types of request templates

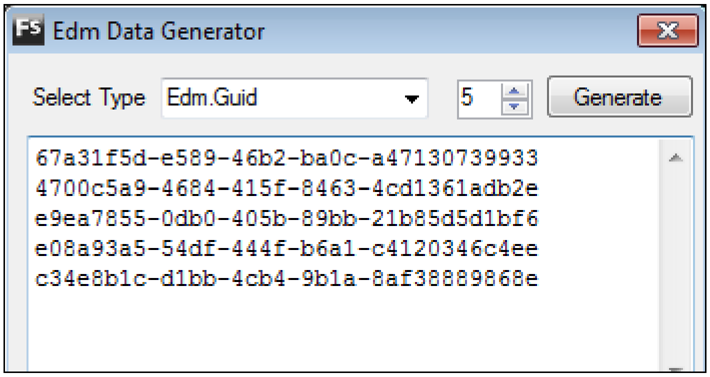


```
POST /Products HTTP/1.1
Host: localhost
Accept: application/json
Content-Type: application/json

{
  "ID": "1992532887",
  "Name": "jtX",
  "Description": "T8Mqd",
  "ReleaseDate": "1595-03-20T04:14:08",
  "DiscontinuedDate": "1454-09-17T11:50:44",
  "Rating": "451586916",
  "Price": "94.76"
}
```

Figure 7: Image shows matching pre-populated data types template

Oyedata also comes with an easy to use built in Data Generator. Just click the "Data Generator" button on the main screen. The data generator can be used to quickly generate valid data types to be used during tests.



Select Type Edm.Guid

5

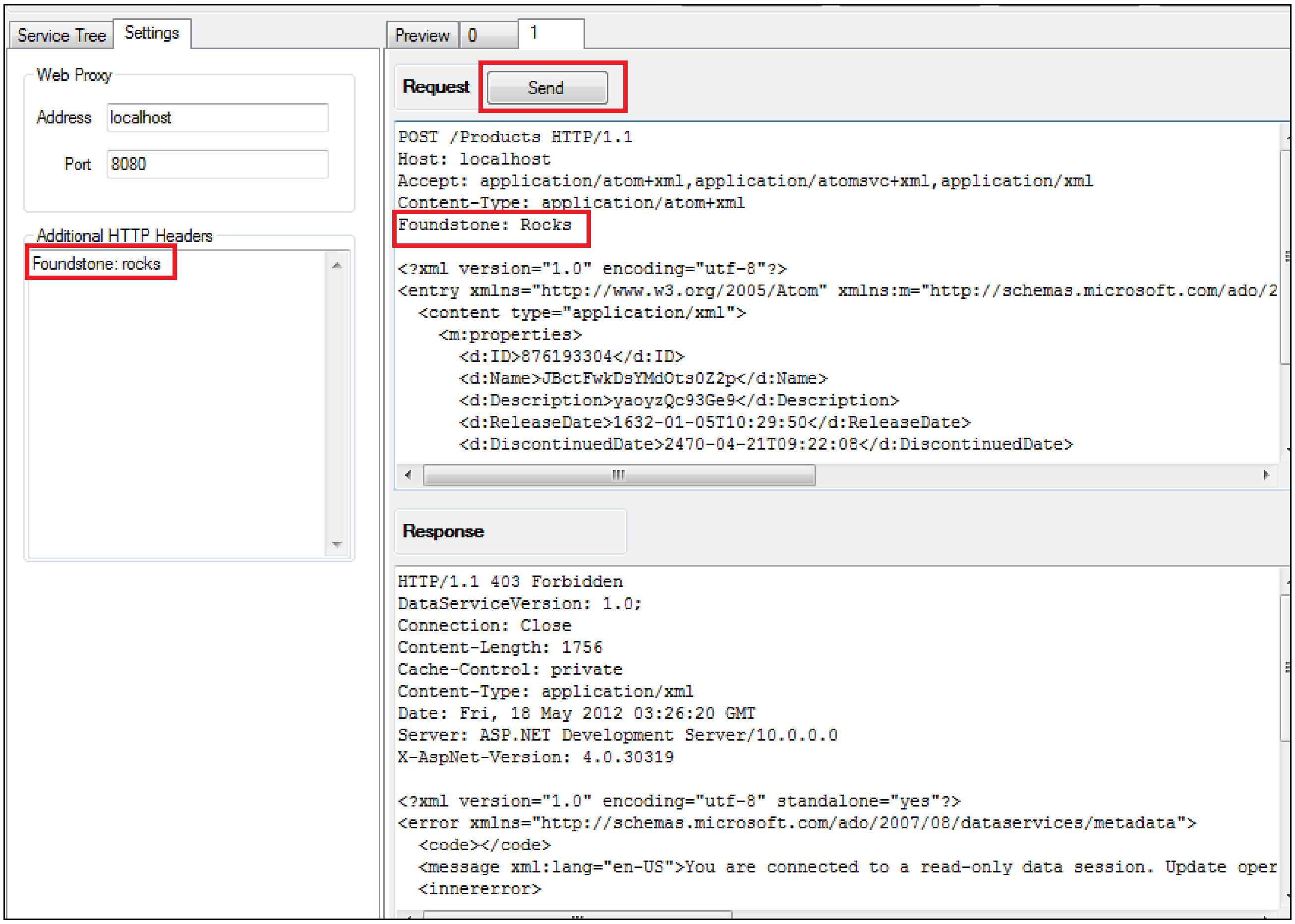
```
67a31f5d-e589-46b2-ba0c-a47130739933
4700c5a9-4684-415f-8463-4cd1361adb2e
e9ea7855-0db0-405b-89bb-21b85d5d1bf6
e08a93a5-54df-444f-b6a1-c4120346c4ee
c34e8b1c-d1bb-4cb4-9b1a-8af38889868e
```

Figure 8: Image shows the Data Generator

Sending Requests

The Send button of the request tab sends the request to the web server. The service response is displayed in the response area in the same tab. Oyedata comes with a couple of additional options that may be useful:

1. The ability to link to an upstream web proxy to be used for sending requests.
2. The option to define custom headers such as Cookies, User-Agents, and additional authentication and authorization information that an OData service may expect.



Web Proxy

Address: localhost

Port: 8080

Additional HTTP Headers

Foundstone: rocks

```
POST /Products HTTP/1.1
Host: localhost
Accept: application/atom+xml,application/atomsvc+xml,application/xml
Content-Type: application/atom+xml
Foundstone: Rocks

<?xml version="1.0" encoding="utf-8"?>
<entry xmlns="http://www.w3.org/2005/Atom" xmlns:m="http://schemas.microsoft.com/ado/2
  <content type="application/xml">
    <m:properties>
      <d:ID>876193304</d:ID>
      <d:Name>JBctFwkDsYMdOts0Z2p</d:Name>
      <d:Description>yaoyzQc93Ge9</d:Description>
      <d:ReleaseDate>1632-01-05T10:29:50</d:ReleaseDate>
      <d:DiscontinuedDate>2470-04-21T09:22:08</d:DiscontinuedDate>
```

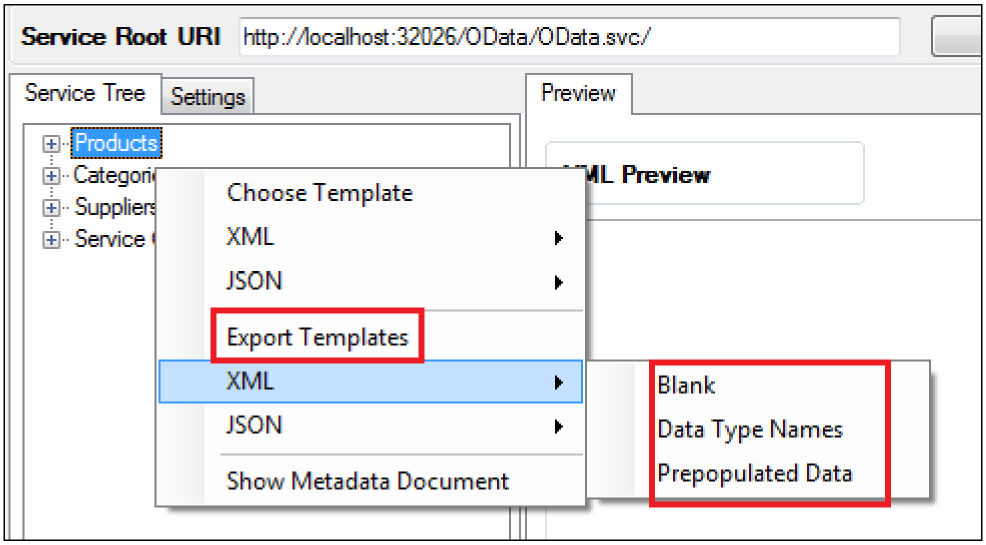
```
HTTP/1.1 403 Forbidden
DataServiceVersion: 1.0;
Connection: Close
Content-Length: 1756
Cache-Control: private
Content-Type: application/xml
Date: Fri, 18 May 2012 03:26:20 GMT
Server: ASP.NET Development Server/10.0.0.0
X-AspNet-Version: 4.0.30319

<?xml version="1.0" encoding="utf-8" standalone="yes"?>
<error xmlns="http://schemas.microsoft.com/ado/2007/08/dataservices/metadata">
  <code></code>
  <message xml:lang="en-US">You are connected to a read-only data session. Update oper
  <innererror>
```

Figure 9: Image shows Oyedata options to engage an OData service and a sample OData service response

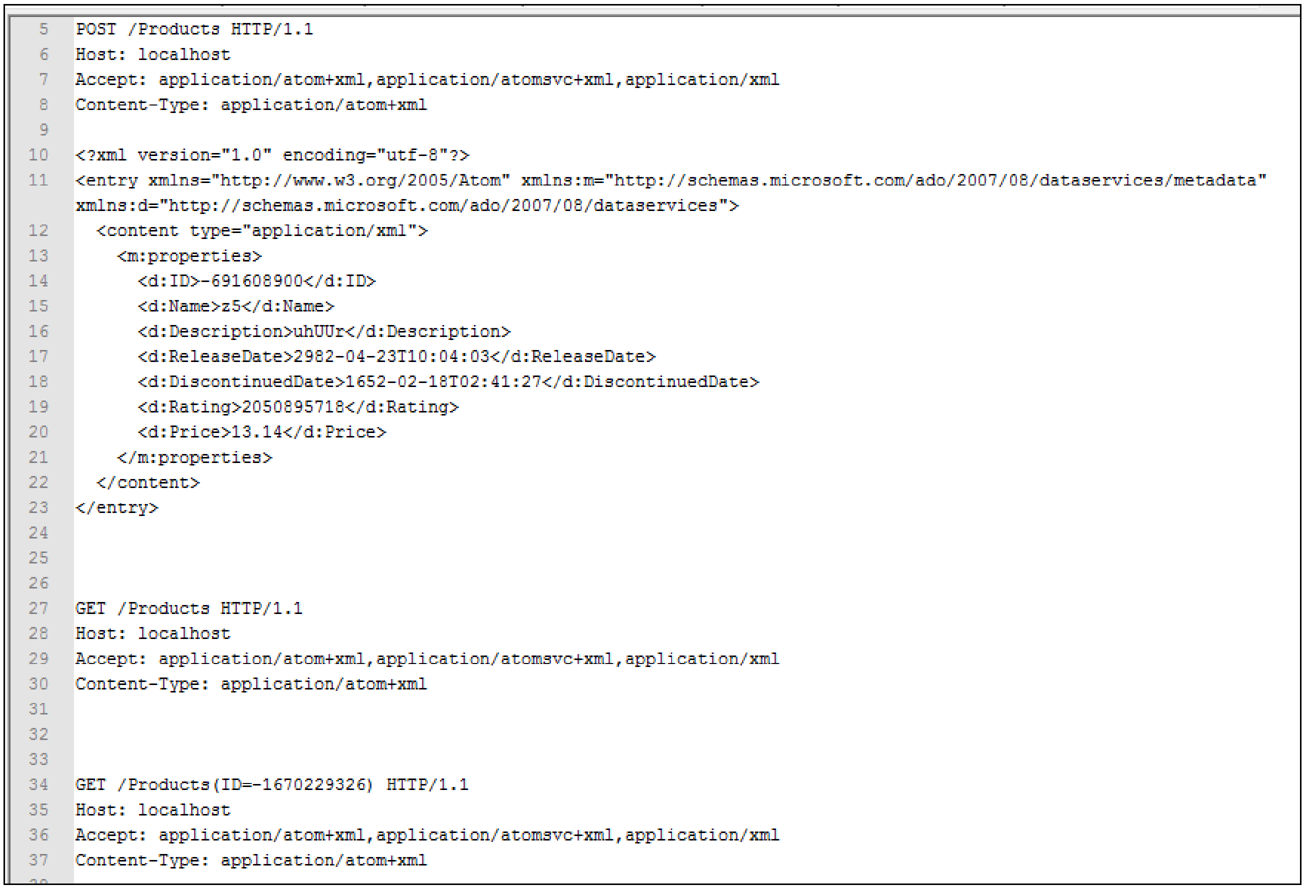
Exporting Attack Templates

To facilitate fuzzing and other attacks, Oyedata provides the option to export JSON or XML attack templates to a text file which can then be supplied to custom fuzzers for automated vulnerability discovery. To export all templates, right click any node, then under "Export Templates", choose to export in either JSON or XML.



Service Root URI http://localhost:32026/OData/OData.svc/

Figure 10: Image shows Export Template functionality



```
POST /Products HTTP/1.1
Host: localhost
Accept: application/atom+xml,application/atomsvc+xml,application/xml
Content-Type: application/atom+xml

<?xml version="1.0" encoding="utf-8"?>
<entry xmlns="http://www.w3.org/2005/Atom" xmlns:m="http://schemas.microsoft.com/ado/2007/08/dataservices/metadata"
xmlns:d="http://schemas.microsoft.com/ado/2007/08/dataservices">
  <content type="application/xml">
    <m:properties>
      <d:ID>-691608900</d:ID>
      <d:Name>z5</d:Name>
      <d:Description>uhUUr</d:Description>
      <d:ReleaseDate>2982-04-23T10:04:03</d:ReleaseDate>
      <d:DiscontinuedDate>1652-02-18T02:41:27</d:DiscontinuedDate>
      <d:Rating>2050895718</d:Rating>
      <d:Price>13.14</d:Price>
    </m:properties>
  </content>
</entry>



GET /Products HTTP/1.1
Host: localhost
Accept: application/atom+xml,application/atomsvc+xml,application/xml
Content-Type: application/atom+xml



GET /Products(ID=-1670229326) HTTP/1.1
Host: localhost
Accept: application/atom+xml,application/atomsvc+xml,application/xml
Content-Type: application/atom+xml
```

Figure 11: Image shows screenshot of exported XML templates

Oyedata – The Road Ahead

Few important Oyedata features that can be seen in upcoming releases are listed below:

1. Add XML and JSON syntax highlighting.
2. Add Media Linked Entries support.
3. Add Batch request processing support.
4. Enhance the template generation algorithm to create update requests templates for individual Entry properties.
5. And more… Stay tuned!

Conclusion

The OData protocol specification does not outline any security considerations and encourages developers incorporate security as per their requirements. Since OData is a new and developing protocol, its implementations may have exploitable issues. Oyedata aims to assist penetration testers and developers to audit their OData implementation before going live.

About The Author

Gursev Singh Kalra serves as a Principal Consultant with Foundstone Professional Services, a division of McAfee. Gursev has done extensive security research on CAPTCHA schemes and implementations. He has written a Visual CAPTCHA Assessment tool TesserCap that was voted among the top ten web hacks of 2011. He has identified CAPTCHA implementation vulnerabilities like CAPTCHA Re-Riding Attack, CAPTCHA Fixation and CAPTCHA Rainbow tables among others. OData security research is one of his interests. He has also developed open source SSL Cipher enumeration tool SSLSmart and has spoken at conferences like ToorCon, OWASP, NullCon, Infosec Southwest and Clubhack.

About Foundstone Professional Services

Foundstone® Professional Services, a division of McAfee. Inc., offers expert services and education to help organizations continuously and measurably protect their most important assets from the most critical threats. Through a strategic approach to security, Foundstone identifies and implements the right balance of technology, people, and process to manage digital risk and leverage security investments more effectively. The company's professional services team consists of recognized security experts and authors with broad security experience with multinational corporations, the public sector, and the US military.