IronWASP (Iron Web application Advanced Security testing Platform)

1. Introduction:

IronWASP (Iron Web application Advanced Security testing Platform) is an open source system for web application vulnerability testing. It is designed to be customizable to the extent where users can create their own custom security scanners using it. Though an advanced user with Python/Ruby scripting expertise would be able to make full use of the platform, a lot of the tool's features are simple enough to be used by absolute beginners.
The major features of this tool are:

- UI based and very easy to use, no much security expertise required.
- Powerful and effective scanning engine with automatic and manual crawling options.
- Supports recording Login sequence.
- We can generate Reports in both HTML and RTF formats.
- Checks for over 25 different kinds of well-known web vulnerabilities.
- False Positives detection support and False Negatives detection support.
- Extensible via plug-ins or modules in Python, Ruby, C# or VB.NET
- Comes bundled with a number of Modules built by researchers in the security community.
- Embedded interactive testing tools to test for:
    1. CSRF Protection
    2. Broken Authentication
    3. Hidden Parameters
    4. Privilege Escalation

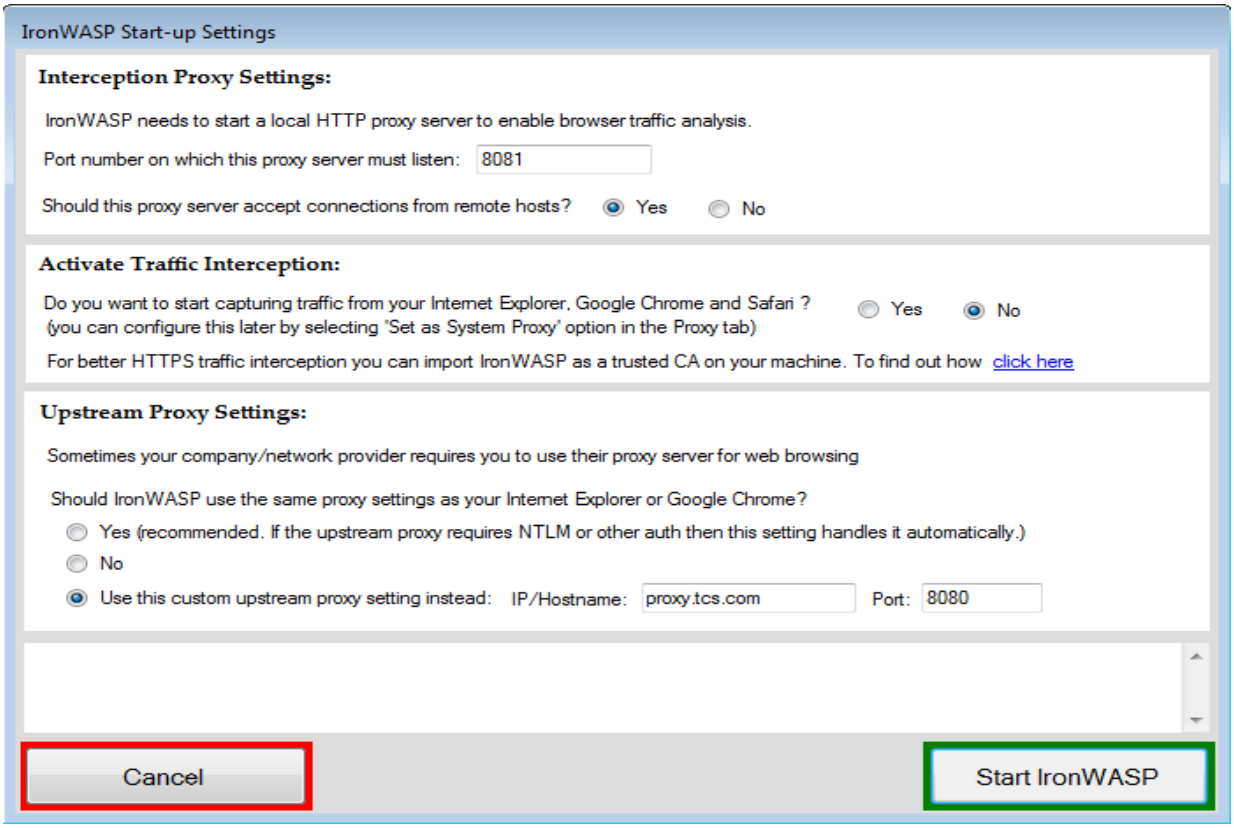
Another major advantage of this tool is that if the user has a very good knowledge in Python or Ruby, they can do a lot more with this tool, like creating their own custom scanners.

You can download the tool from the given link http://ironwasp.org/download.html

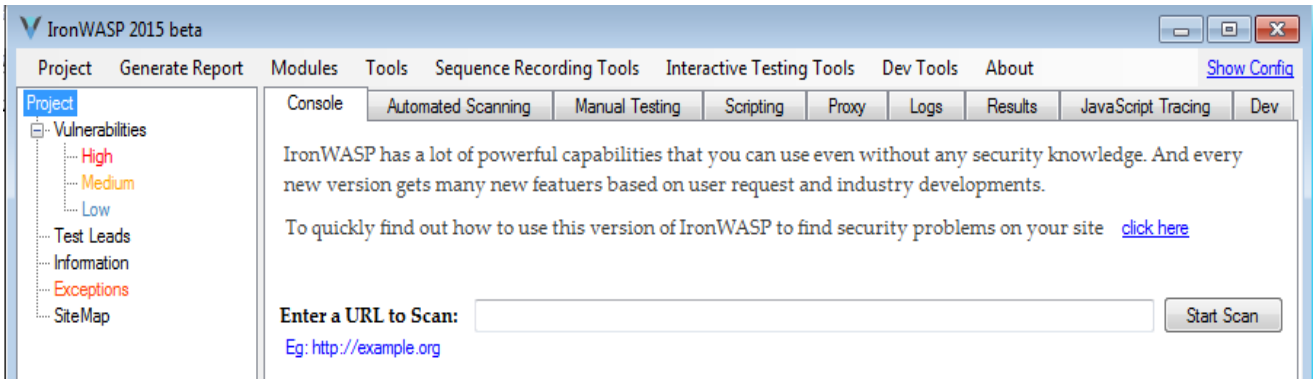
2. Getting Started:

Once you have downloaded the setup and extracted the files to your desired location, double-click on the IronWASP application found inside the IronWASP folder.

Double-click the icon and provide the initial proxy details at the application start-up wizard and select start button as shown below



Once started the we will get the console as shown below:



3. Performing Vulnerability Scans with IronWASP:

We can scan the application from IronWASP tool in two different ways:

Automatic scanner and Crawling the application & scanning the result

3.1 Automatic scanner:

You can directly provide the URL of the application you intend to scan in the 'Enter a URL to scan' section in console tab (Fig 3.1.1)and select start scan which will open up the scan creation wizard (Fig 3.1.2)

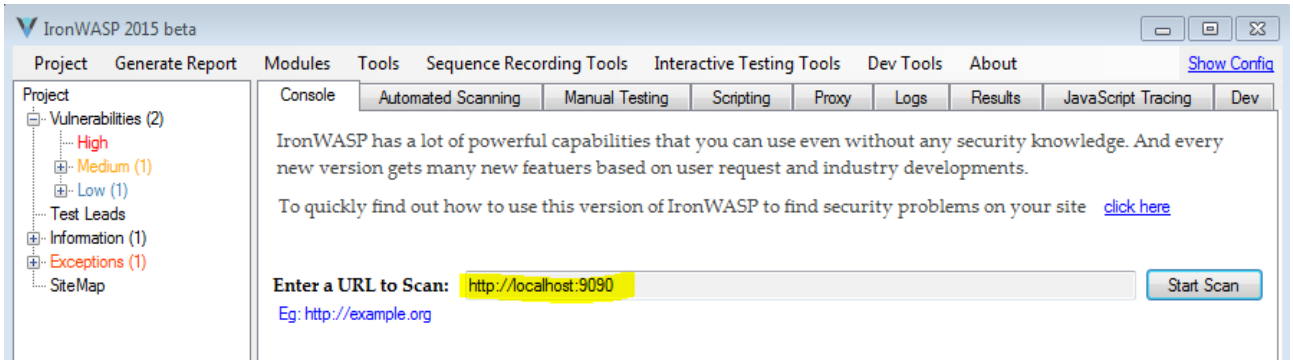


Fig 3.1.1 Entering URL in the console

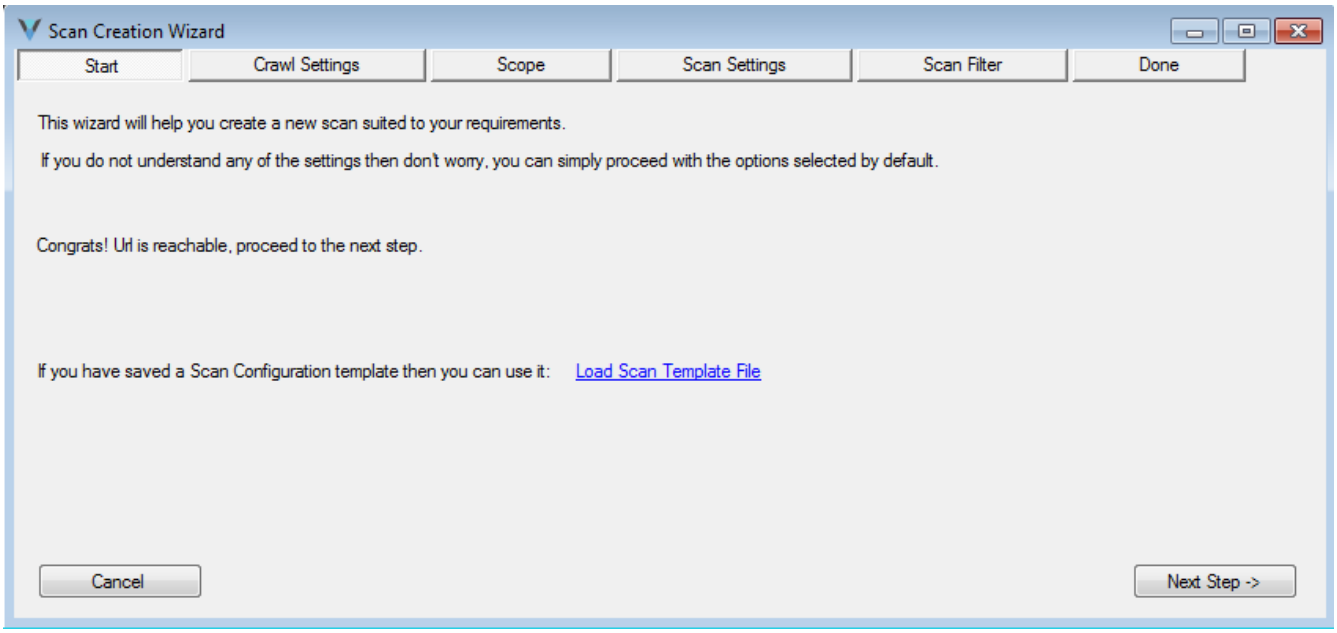


Fig 3.1.2 Scan creation wizard

Update the settings as per the requirement start the scan and we can see the information about scan jobs created, Jobs completed & vulnerabilities found in the console as shown in the fig 3.1.3 below:

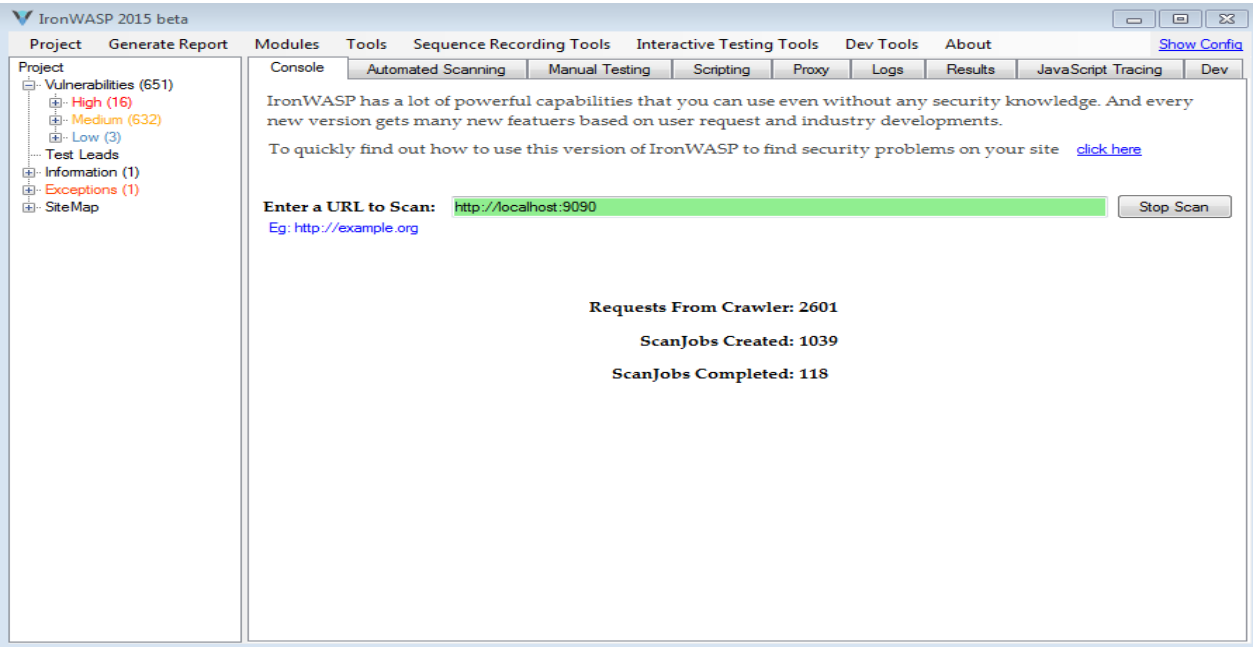
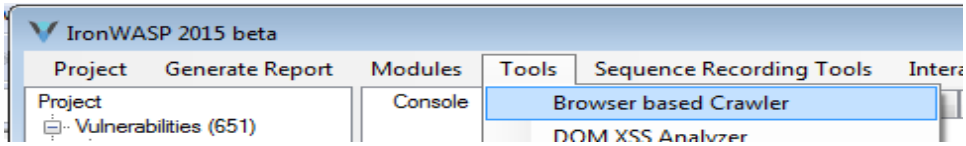


Fig 3.1.3 Details of the scan

3.2 Crawling the application & scanning the result:

We can you the browser based crawler wizard for effective coverage of the application scope.

To open the browser based crawler- go to tools tab and click on 'Browser based crawler'.



Once selected browser based crawler wizard will open as shown in the fig 3.2.1 below:

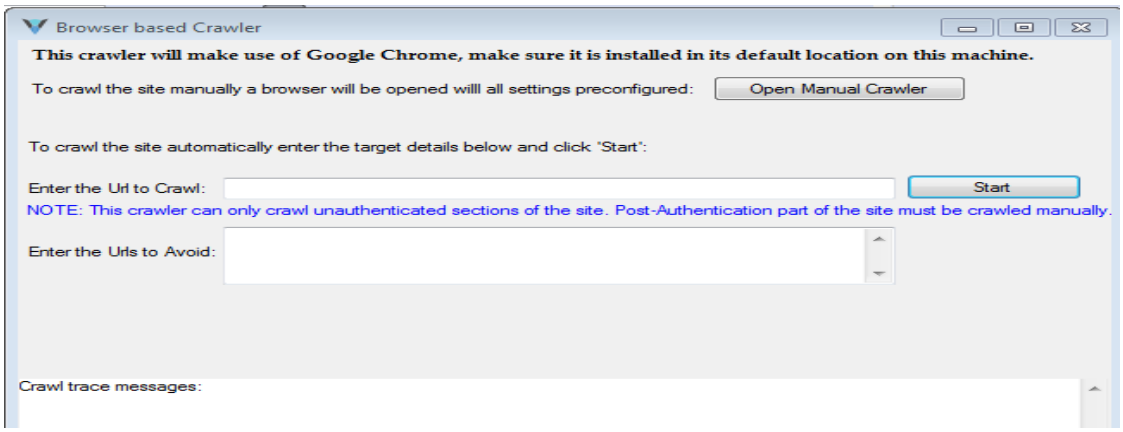
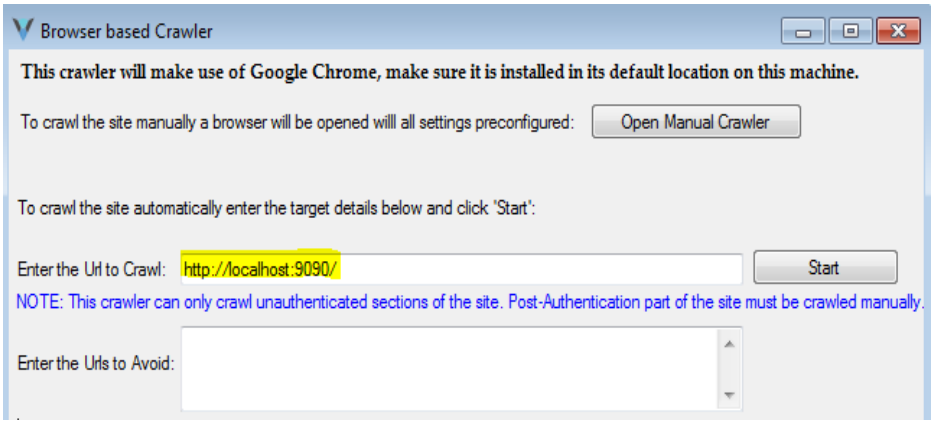


Fig 3.2.1 Browser based crawler wizard

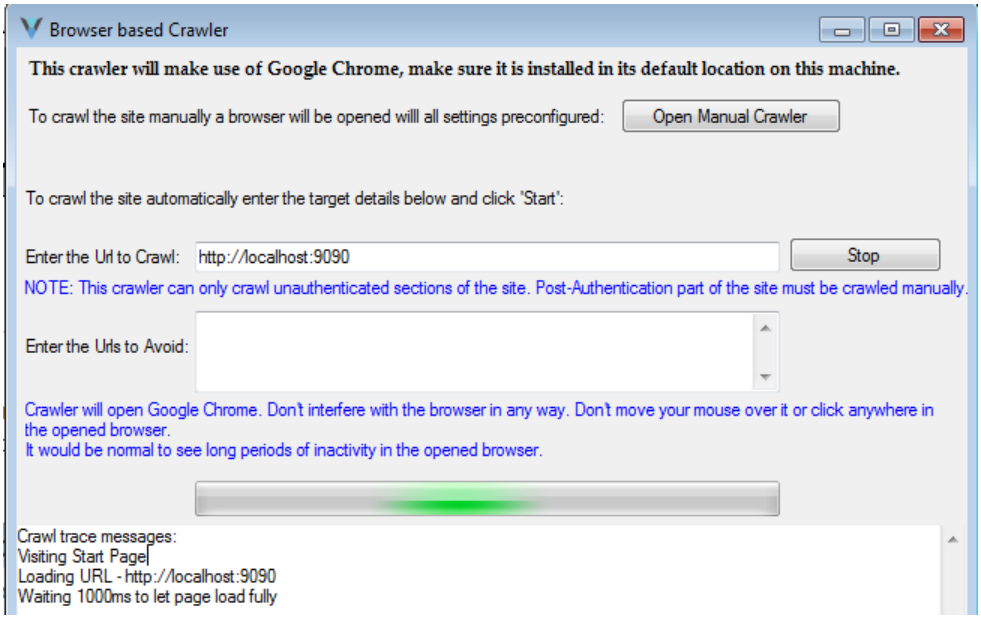
Here we have two options to crawl the application: Automatically crawl and manually crawl.

1. Automatically crawl the application:

Provide the URL of the application in the 'enter the URL to crawl section of the wizard and click on start as shown in the figure 3.2.2:

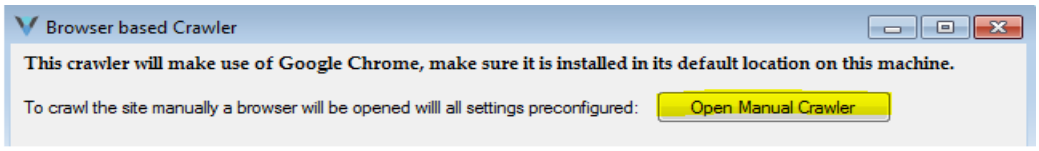


This will crawl the application more effectively as it uses the browser to crawl the application and crawling details will be shown in the fig 3.2.3 below:

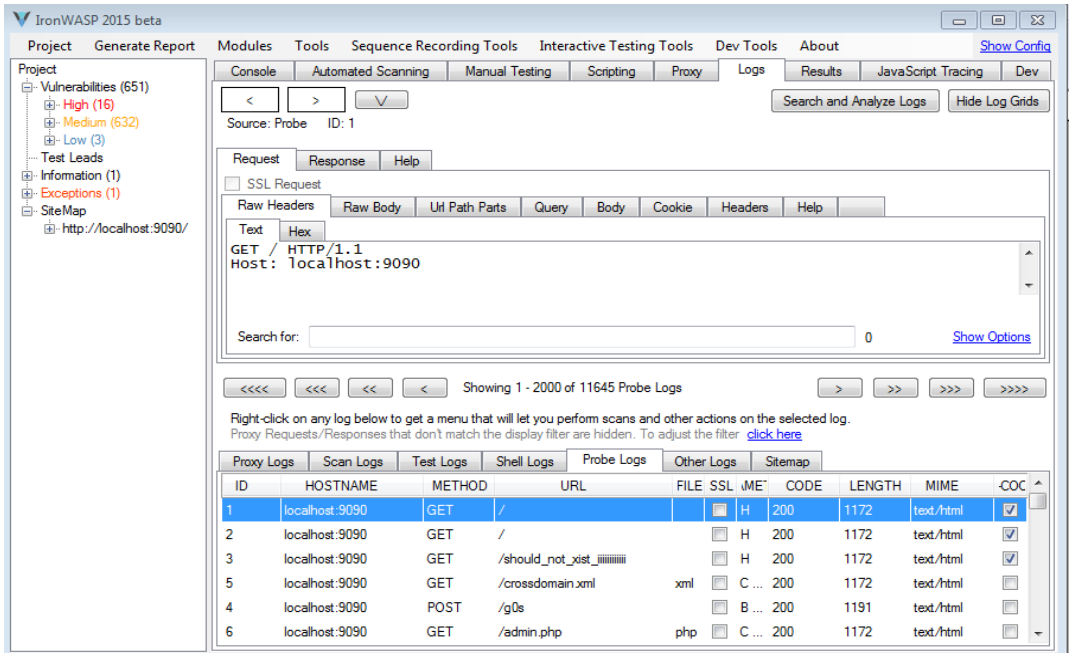


2. Manually crawl the application

Here select the open manual crawler option in the browser based crawler wizard, which will open the browser where we can manually crawl the application which is very useful for complex application with multiple login pages



Once we are done with crawling the application; tool will capture all the logs which we can view in the logs tab as shown in the fig 3.2.4 below:



here we can run the scan per request (Fig 3.2.5) and we can run the scan for the whole site (Fig 3.2.6) as shown below:

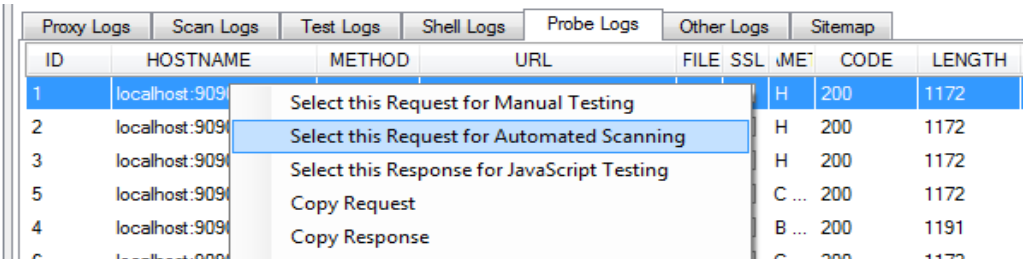


Fig 3.2.5 Scanning each request from logs tab

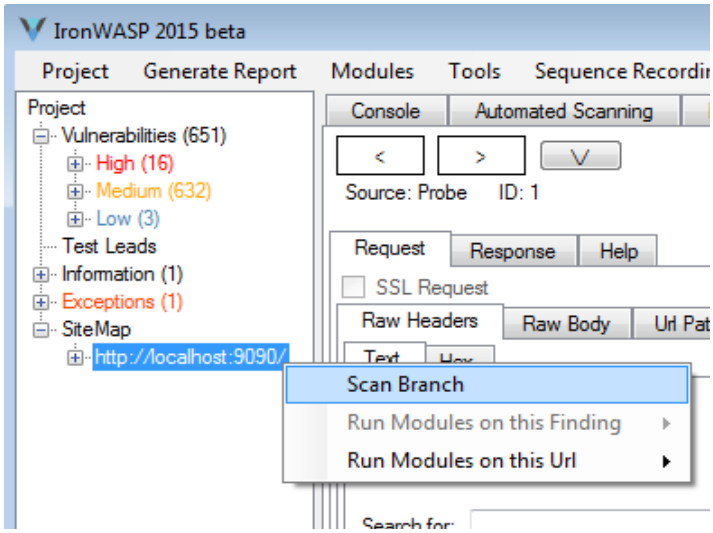
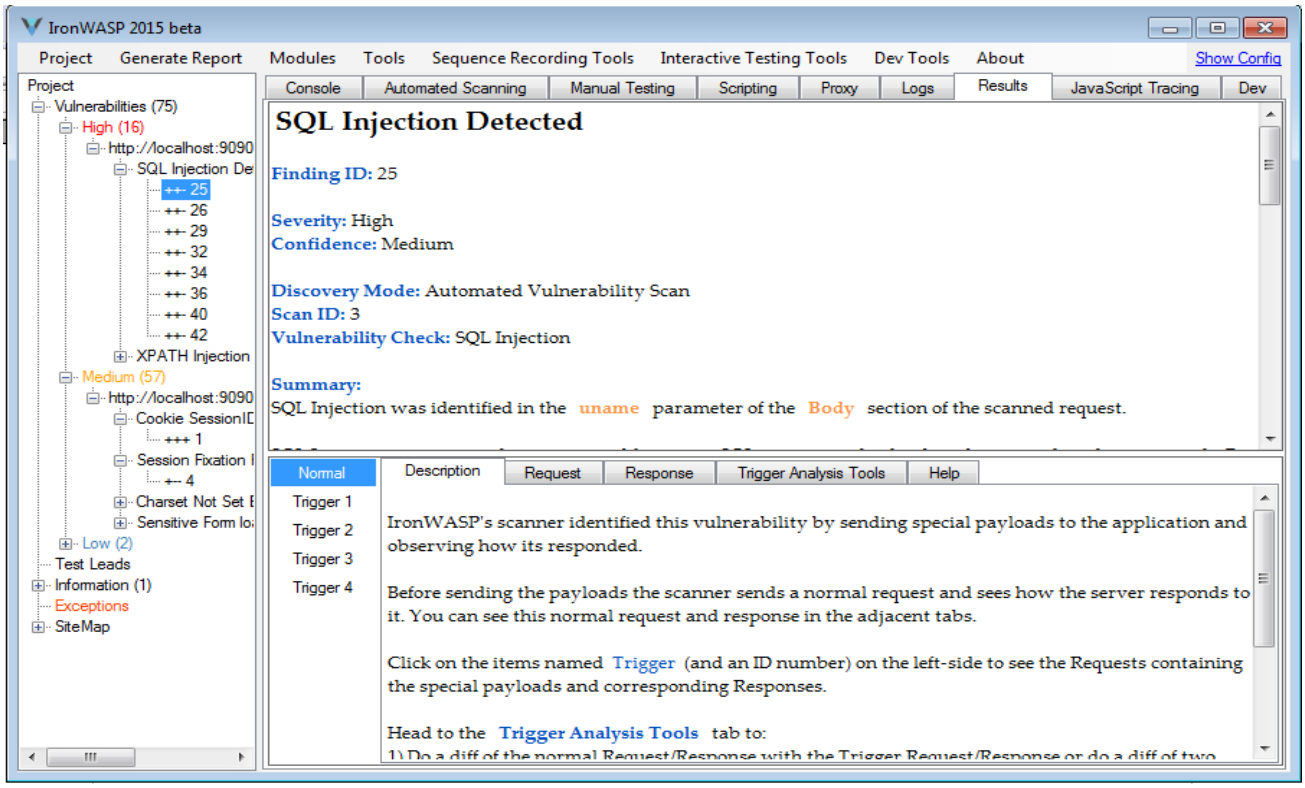


Fig 3.2.6 scanning the whole site from site map

4. Scanning result information and Report generation

4.1 Scanning result information:

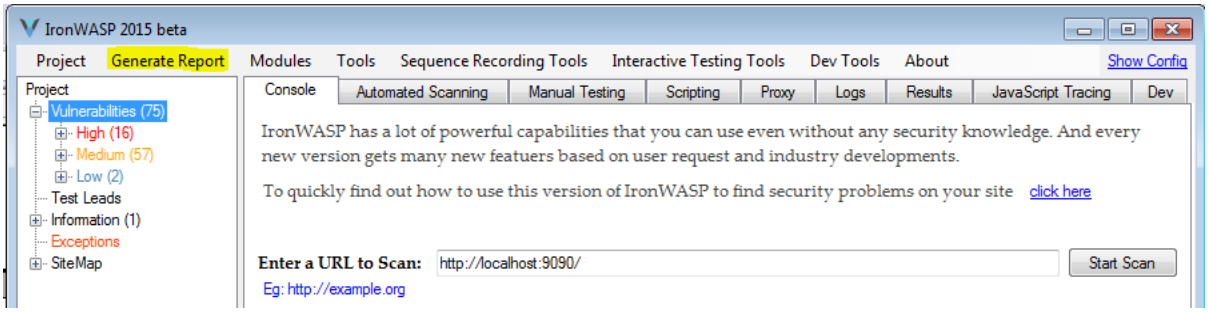
Once the vulnerabilities get detected, they are classified as High, Medium, or Low, depending on the impact. We can find the vulnerability details in the left hand side if the console and on clicking on each issue, we will get the full details like description, payload information etc. as shown in the fig 4.1 below



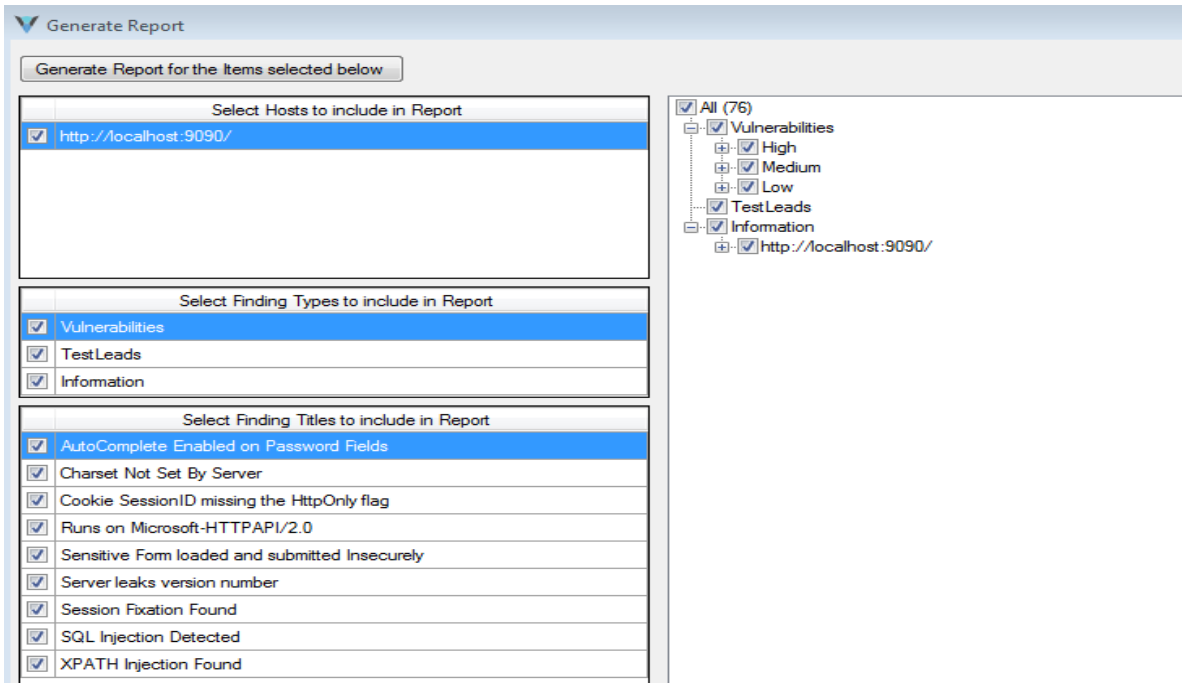
4.2 Report generation

IronWASP tool will generate reports both in HTML and RTF formats, report generation steps are as explained below:

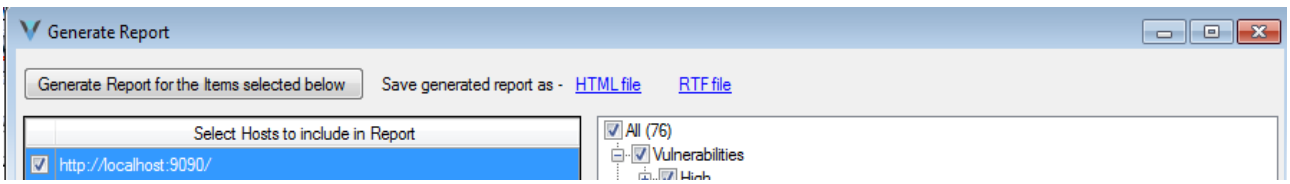
Select the 'Generate report' tab as shown in the fig 4.2.1 below:



On selecting report generation wizard will open as shown in the Fig 4.2.2



Select all the required information for the report and click on 'Generate report for the items selected below', which will generate the report and provide options to save in HTML or RTF format.



5. Other Major features:

5.1 Plugins

IronWASP has a plugin system that supports Python and Ruby. The version of Python and Ruby used in IronWASP is IronPython and IronRuby which is syntactically similar to CPython and CRuby. However some of the standard libraries might not be available, instead plugin authors can make use of the powerful IronWASP API.

- The Github repository of the Ruby plugins
- The Github repository of the Python plugins

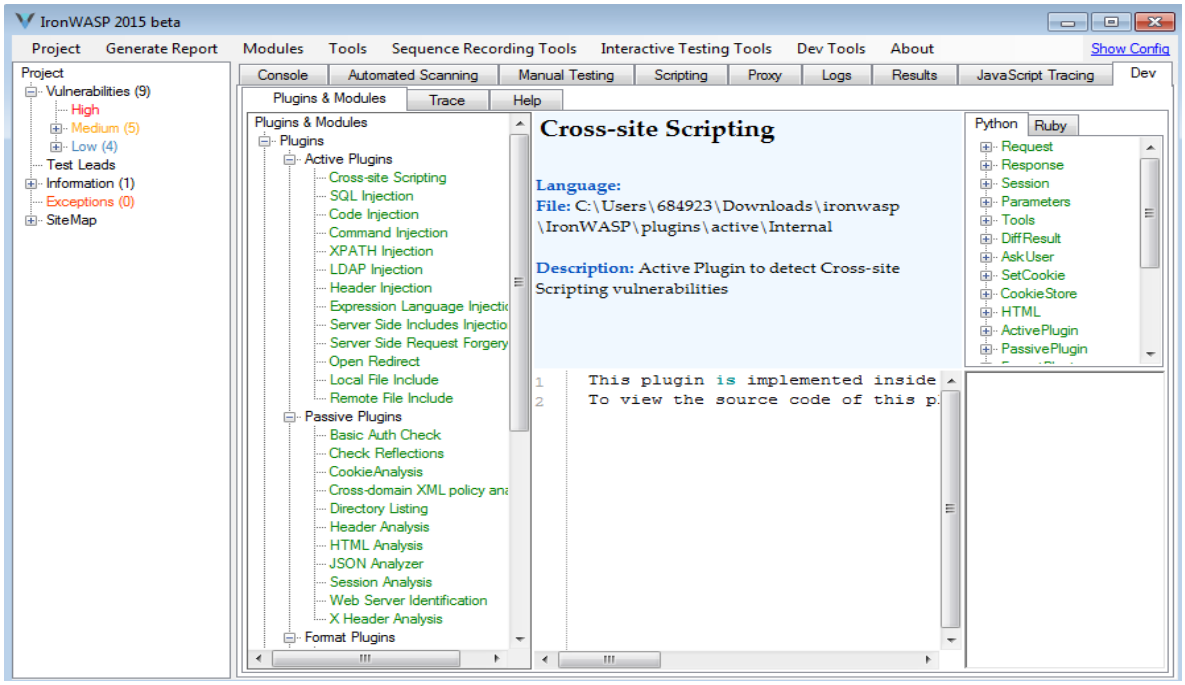
1. Passive Plug-ins:
   - Analyzes all traffic going through the tool
   - Can also modify the traffic
   - Identifies vulnerabilities passively

Eg: Passwords sent over clear-text
Http-Only /Secure flag missing in cookies

2. Active Plug-ins:
   - Performs scans against the target to identify vulnerabilities
   - Executed only when the user explicitly calls them
   - Fine-grained scanning support

Eg: Cross-site Scripting
SQL Injection

3. Format Plug-ins
   - To deal with various data formats in Request body.
   - Allows scanning even for custom data formats



5.2 Integrated Scripting Engine:

- Made of Python/Ruby scripts using IronWASPAPI
- Full access to all the functionality of the tool
- Can create precise Crawlers and Scanners
- Can analyze the HTTP logs for Access Control and other checks
- Extremely simple and easy to use

Some of the Available Classes are:

- Request
- Response
- IronSession
- Crawler
- Scanner
- Tools
- HTML

5.3 JavaScript Static Analysis:

-
- IronWASPperforms Taint Analysis for DOM based XSS
- Identifies Sources and Sinks and traces them through the code
- Custom Source and Sink objects can be configured
- Handles a few JavaScript quirks likea.b being presented as a[“b”]or varx = “b”; a[x]

Conclusion:

IronWASP is an environment for web application security testing designed for optimum mix of manual and automated testing. It has a GUI interface which doesn’t require any installation and comes with Built-in Crawler, Scan Manager & Proxy and embedded with modules & plugin's. IronWASP is able to detect most of the vulnerabilities with least number of "false positives" and lets us write a custom Security Scanner in a very short time.

References:

https://ironwasp.org/index.html
https://ironwasp.org/learn.html
https://ironwasp.org/about.html