

Web Profiling and Scoping Tool

Technical Document

Indiana University, Bloomington

Computer Science

Spring Semester, 2016

**Team Members**

Abhijit Karanjkar

Ajinkya Dhamnaskar

Amruta Kamat

Gagandeep Singh

Pranav Pande

**Table of Contents**

[Tool Summary](#h.txijn4lyz0et)

[About Tool](#h.xrlyul34ztl8)

[Tool Architecture](#h.gi9i9hhzcpdr)

[Technologies Used](#h.2vsvhrbyb4hy)

[Functionalities](#h.igbkzsje6l1q)

[User Inputs](#h.l18eu1wk7oou)

[Active Functions](#h.v250909hakvn)

[Host Information](#h.38fxj3ojasd0)

[Port Scanner](#h.ikvysban8bt6)

[Web Crawler](#h.7lwk3enm75lu)

[Passive Functions](#h.z1cno4gj7u51)

[Vulnerabilities](#h.7s9ncaxx3s5t)

[Header Information](#h.vpkdwdhccuyd)

[Cookie Information](#h.f1rujfmxj0u9)

[Technologies Used](#h.97iqtevhyd1e)

[Edit JSON](#h.q229gkd86skm)

[Additional Features](#h.6i1tyv16u40j)

[Download JSON](#h.y2fayomxdo0z)

[Clear all](#h.sb7p52z2uom)

[Local db](#h.qyp11dgxto7o)

This document will provide details regarding the technical details of each functionality. The technical details includes information regarding the files, functions and objects associated with each feature.

# Tool Summary

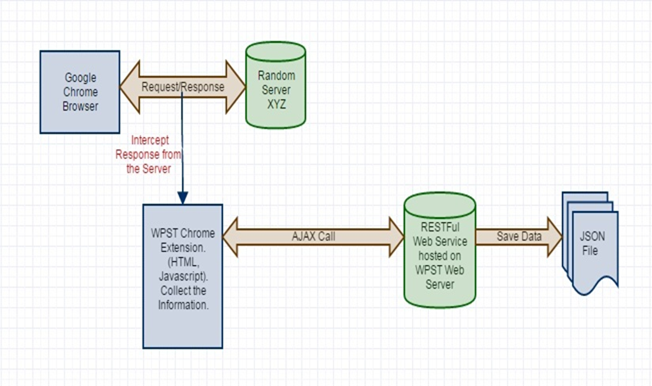
## About Tool

This tool is a chrome extension can be used for retrieving following information regarding any web application on chrome browser. The information which will be retrieved includes:

1. User inputs about the web application
2. Host IP
3. Ports used
4. Available web links
5. Vulnerabilities
6. Header Information (Before and after login if login functionality available)
7. Cookies Information (Before and after login if login functionality available)
8. Technologies of the application

The user of the tool can download all this information in a JSON file.

## Tool Architecture



## Technologies Used

* + 1. Languages

1. JavaScript
2. Java (for Restful Web Services)
   * 1. Open Source Network Security tools
3. Wappalyzer
4. Retire
5. Site Spider

# Functionalities

The main page of the tool will be rendered from *ProfilingChromeExtension\form.html* file. Once the tool is opened and *ProfilingChromeExtension/WebProfiling/js/custom/loginPopup.js* will be loaded which generates a popup for getting user input regarding login information.

## User Inputs

Files:

1. *ProfilingChromeExtension\form.html*
2. *ProfilingChromeExtension\js\custom.js*

Object for storing information:

1. userInputMap

## Active Functions

## Host Information

Files:

1. *ProfilingChromeExtension\form.html*
2. *ProfilingChromeExtension\js\custom.js*

Object for storing information:

1. dnsMap

Function used:

1. getHostInfo()

## Port Scanner

Files:

1. *ProfilingChromeExtension\form.html*
2. *ProfilingChromeExtension\js\custom.js*

Object for storing information:



Function used:

1. testOpenPorts()

## Web Crawler

Files:

1. *ProfilingChromeExtension\form.html*
2. *ProfilingChromeExtension\js\custom\spider.js*

## Passive Functions

## Vulnerabilities

Files:

1. *ProfilingChromeExtension\js\retirejs*

Javascripts under this directory

Object for storing information:

1. retireJsResult

Function used:

## Header Information

The login information popup which appears on the launch of the tool has three buttons - Yes, No, Not Available. If use clicks No/Not Available then header info will be stored as data before login and this data will also be preserved in local storage of the browser. Before storing data to local storage, it will cleared first. When user clicks Yes then header info will stored as data after login and this time *headerInfo* object contain data before and after login. The data before login will be fetched from local storage.

Files:

1. *ProfilingChromeExtension\form.html*
2. *ProfilingChromeExtension\js\custom\cookieAndHeaderInfo.js*

Object for storing information:

1. headerInfo

Function used:

1. getHeaders()

## Cookie Information

The login information popup which appears on the launch of the tool has three buttons - Yes, No, Not Available. If use clicks No/Not Available then cookie info will be stored as data before login and this data will also be preserved in local storage of the browser. Before storing data to local storage, it will cleared first. When user clicks Yes then cookie info will stored as data after login and this time *cookieInfoJSON* object contain data before and after login. The data before login will be fetched from local storage.

Files:

1. *ProfilingChromeExtension\form.html*
2. *ProfilingChromeExtension\js\custom\cookieAndHeaderInfo.js*

Object for storing information:

1. cookieInfoJSON

Function used:

1. getCookieInfo()

## Technologies Used

Files:

1. *ProfilingChromeExtension\js\wap*

Object for storing information:



Function used:

## Edit JSON

Files:

1. *ProfilingChromeExtension\form.html*
2. *ProfilingChromeExtension\js\custom\jasontohtml.js*

## Additional Features

## Download JSON

Files:

1. *ProfilingChromeExtension\form.html*
2. *ProfilingChromeExtension\js\jasonUtility*

## Clear all

Files:

1. *ProfilingChromeExtension\form.html*
2. *ProfilingChromeExtension\js\localdb.js*

## Local db

Files:

1. *ProfilingChromeExtension\js\localdb.js*