iGNITE Technologies

Burp Suite for Pentester

Active Scan++

WWW.HACKINGARTICLES.IN

Contents

Introduction

Using Burp Suite as an automated scanner? Wondering right, even some pentesters do not prefer it, due to the fewer issues or the vulnerabilities it carries within. But what, if the burp scanner itself could identify the least common vulnerabilities along with core findings.

one of the most popular burp plugins **"Active Scan++"** thereby merges up with the burp's scanner engine in order to enhance its scanning capabilities to identify the additional issues within an application.

Exploring & Initializing Active Scan++

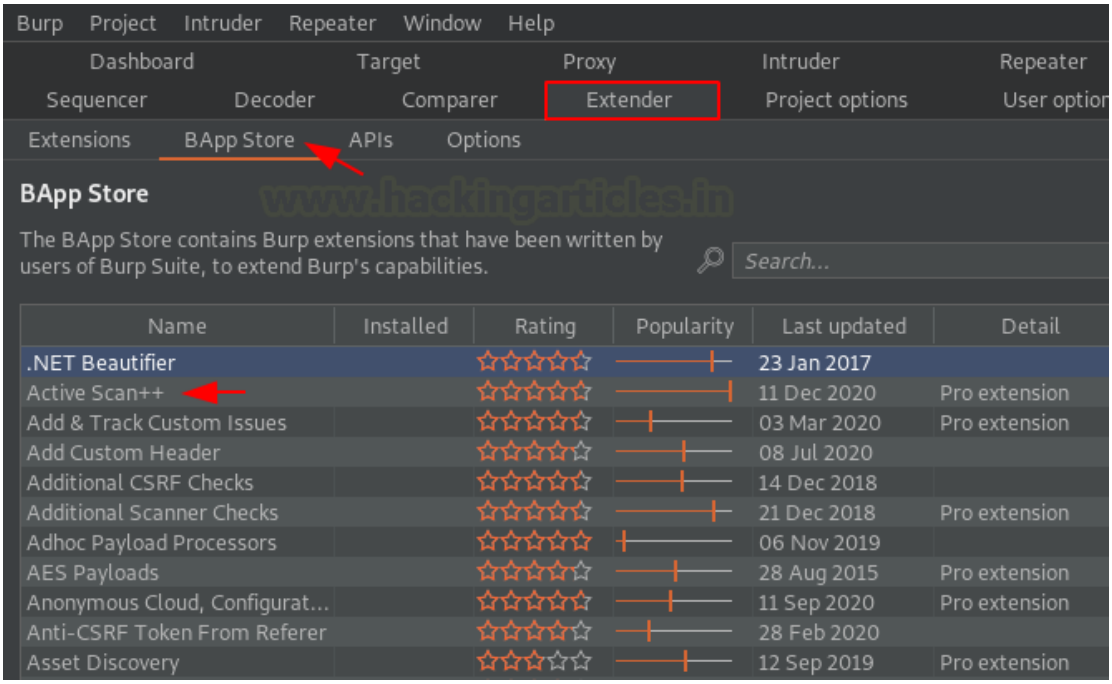
Advanced vulnerabilities require advanced scanning techniques.

Thereby, Active Scan++ is one of the most of most popular burp extensions designed for the **Burp's Professional users** by **"James Kettle"** in order to improvise the burp's active and passive scanning capabilities.

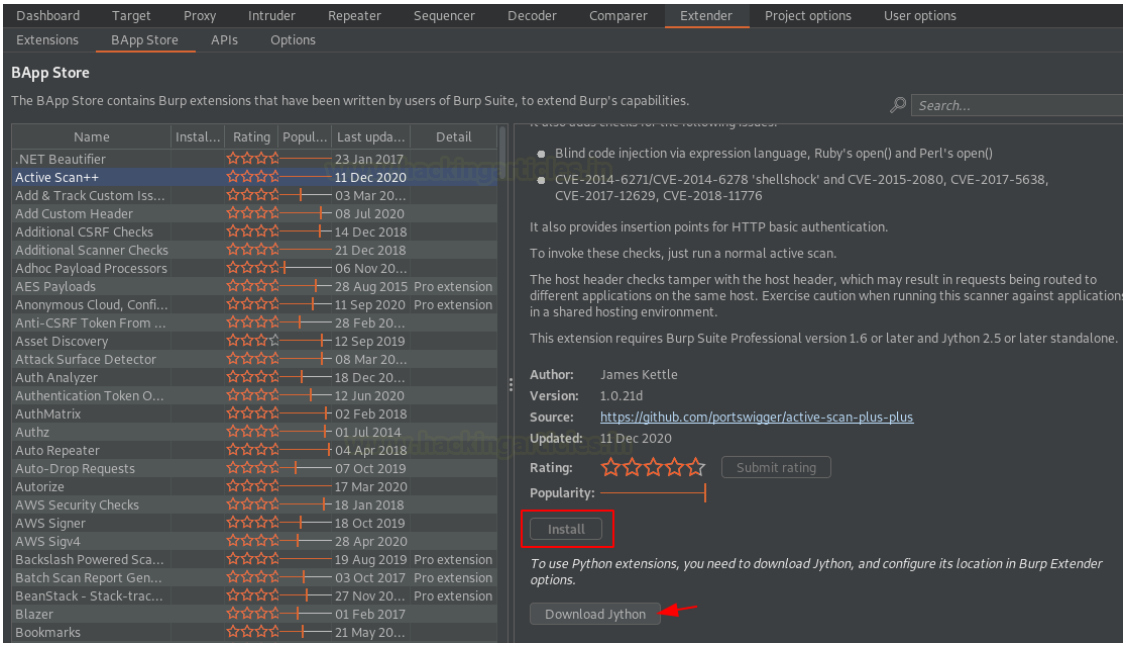
However, this plugin gets integrated within the burp scanner such that it could help in the issue discovering part for the **Host Header Attacks, Password Reset Poisoning, Cache Poisoning, DNS Rebinding, XML Injection, Arbitrary Header Injection, Template Injection, Blind Code Injection**, and the list goes on.

Moreover, this plugin also identifies the insertion points for **HTTP Basic Authentication.**

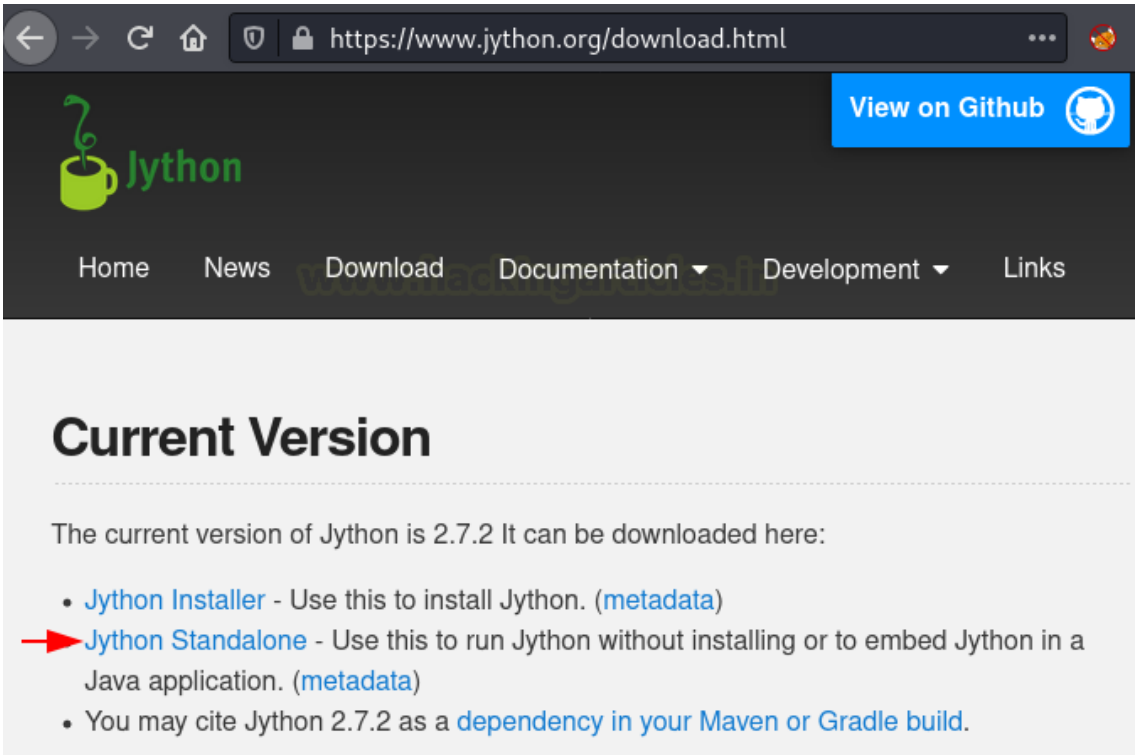
Being so much effective, so let's find it out at the **bApp** store first. And we know where to navigate for that. Over at the **Extender** section, switch to the **bApp store** and then you'll find this tool at the top with the highest rating.



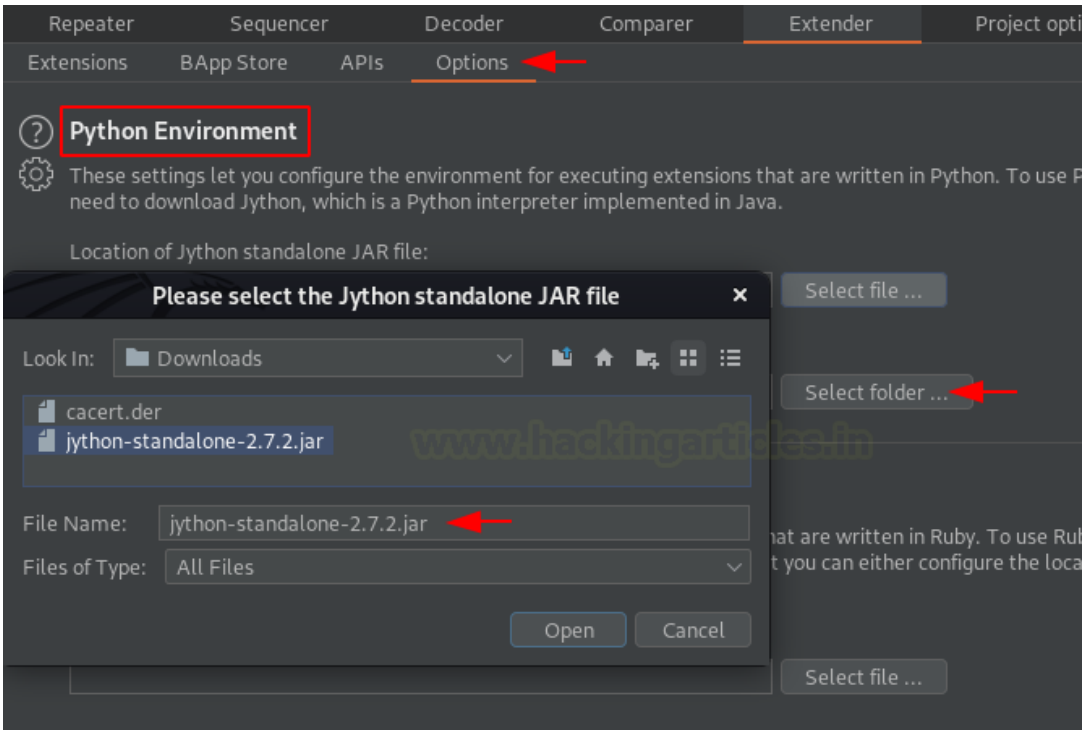
Let's now switch to the left panel in order to identify the **Install** button. But wait!! It requires Jython, so let's install and configure that first.



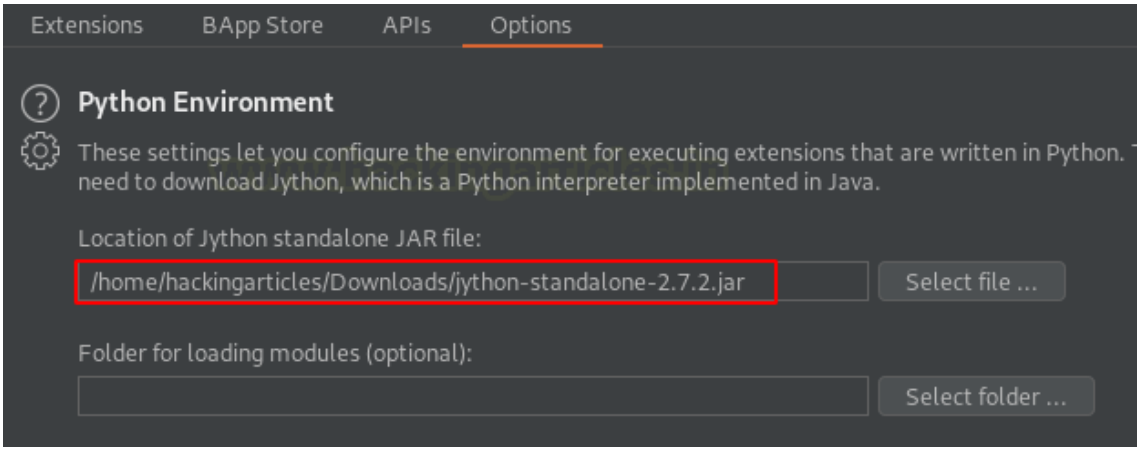
Head to Jython's official website, download the **Standalone Jython's** file.



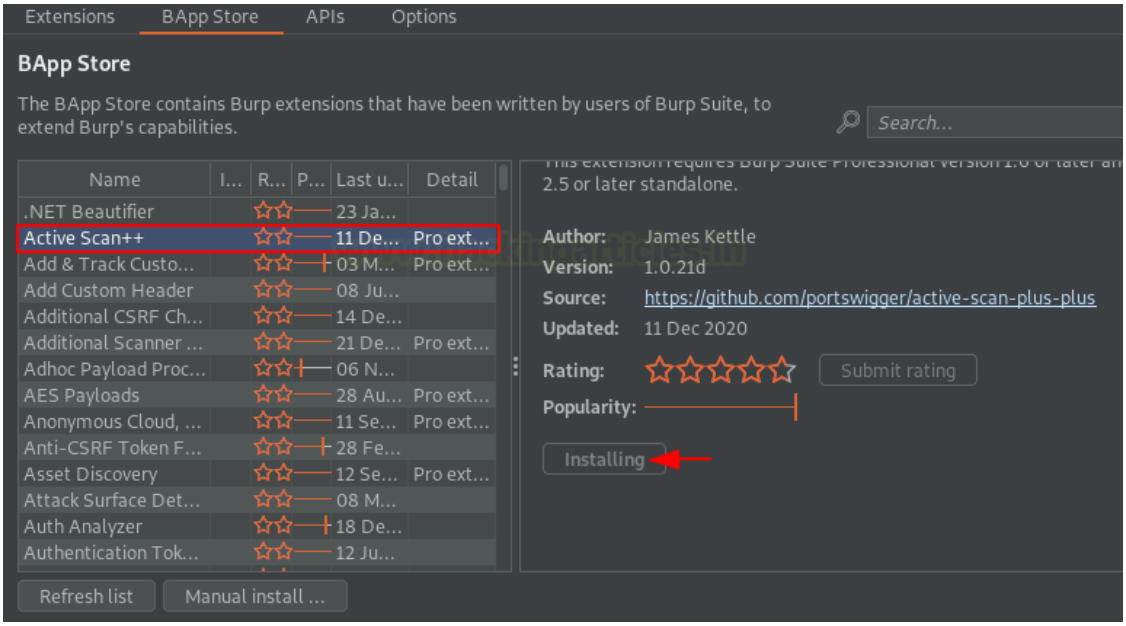
As soon as the file got stored up over at the local machine, we'll embed it with our Burp Suite application. Back at the extender tab, navigate to the **Options** section there and scroll down for the **"Python Environment",** hit the **select file** button, and then opt for the downloaded file.



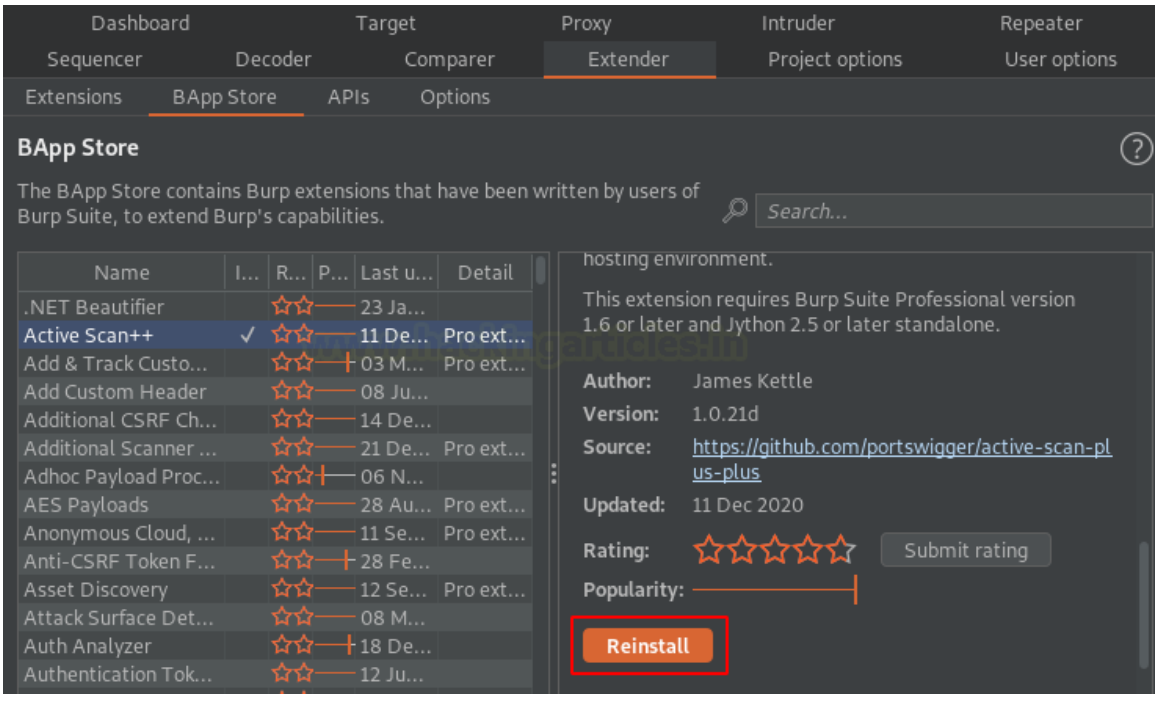
Once done with all this, you'll have your screen somewhat similar to the below image. But, the Jython's configuration is yet not over, **restart your burp** in order to get the changes reflected.



Now head back to the bApp store and open the **Active Scan++** righ-side portal. From the below image you can see that the **Install** button is now active, let's fire it to initiate the installation.



Great!! We got the **Reinstall** button, seems like the extension had been set up successfully.



Enhancing the Audit Functionalities

You might be wondering, like being the most popular, Active Scan++ should have its own place at the top panel, so where it is?

As discussed earlier that **Active Scan++ integrates with the burp's scanner** in order to assist it to identify additional vulnerabilities. Thereby, we do not have any specific location to find it. However, we can analyze its working while performing an active or passive scan.

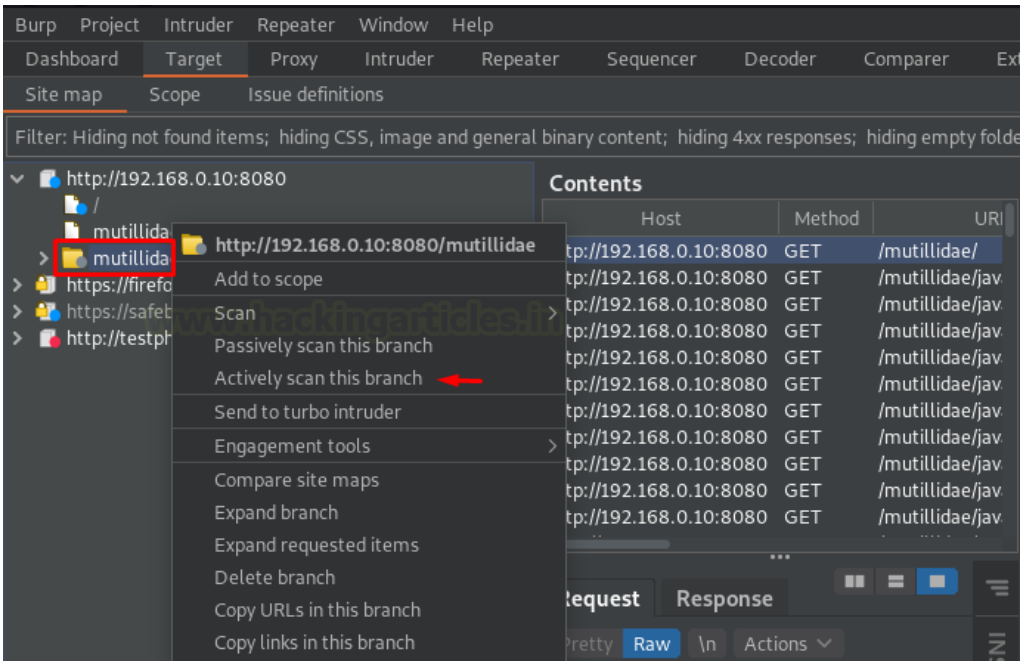
So, let's see what additional it dumps out when we initiate a scan over at the entire application.

Audit the application

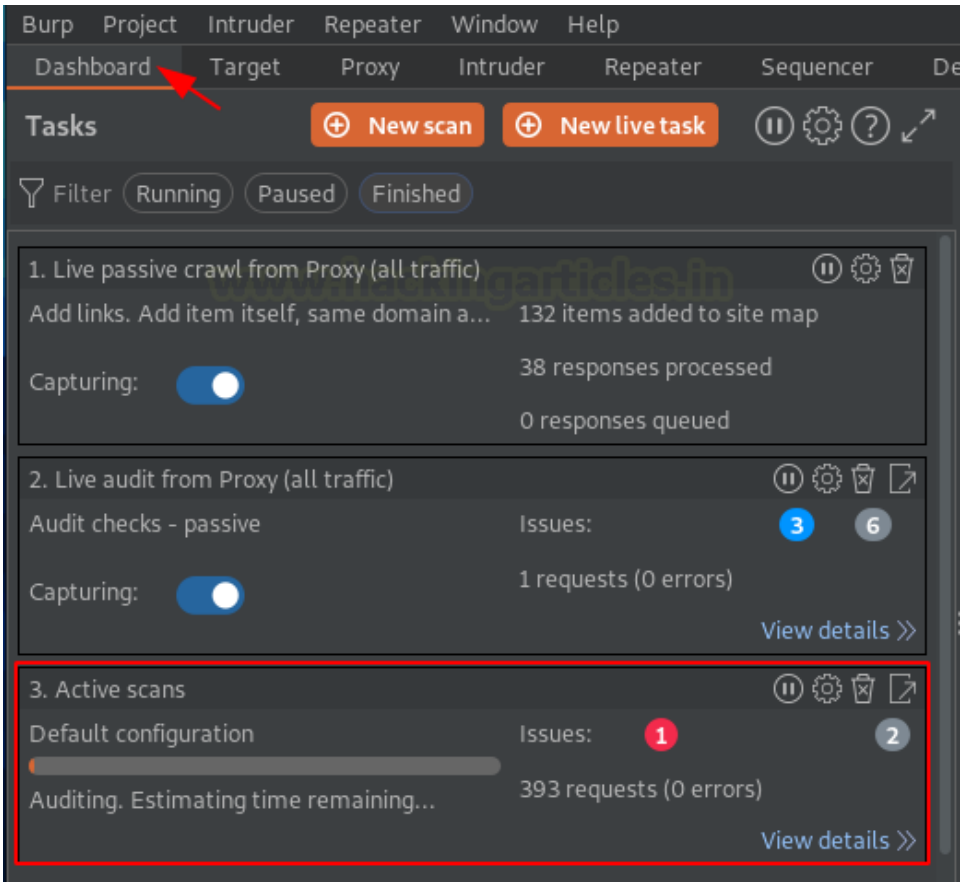
Turn **On** the Browser's **Proxy Service** and then surf the OWASP's Mutillidae vulnerable application.

Now, let's navigate to the **Target** option over at our burp suite monitor, further head to the **Site map**.

The Left panel carries up the web application's hierarchy, let's opt for the **root branch**, and then we'll hit a right-click to opt **"Actively scan this branch"** in order to share the application for the scanning part.

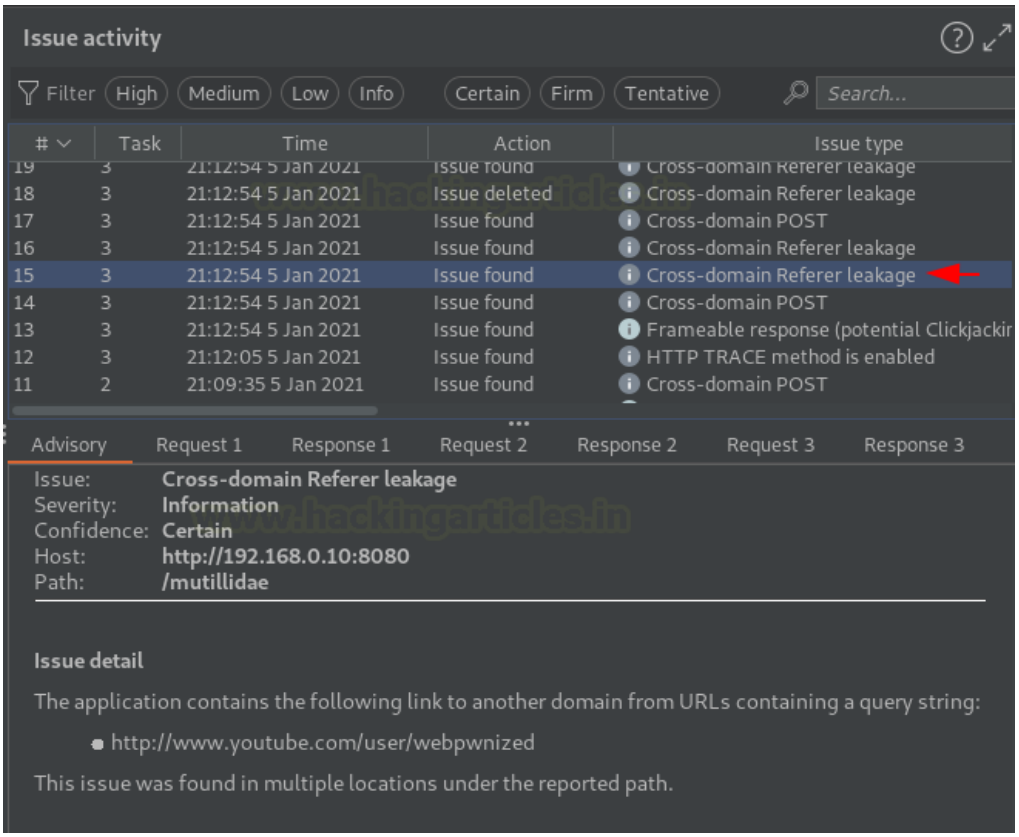


As soon as do so, we got our auditing task aligned at the **"Dashboard".**



And within a few minutes, you can see a number of issues segregated at the right panel, let's check them out.

You might find most of the issues discovered with the burp's scanner but there are some like cache poisoning, DNS rebinding, Host header Injection, all these additional ones are identified with the Active Scan++.



Auditing specific Injection Points

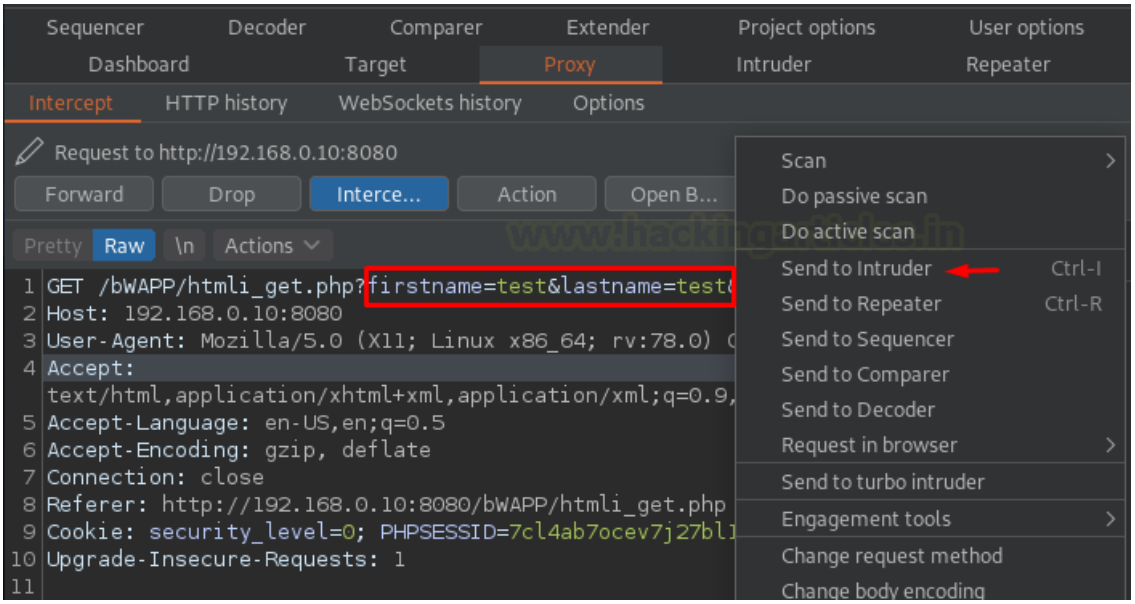
What if, you don't want to test the entire application's branch or a specific web page, but you want an injection point to be audited and if it's possible then does ***Active Scan++*** still collaborate with the burp scanner?

As soon as we hit the install button at the bApp store, the very first-second Active Scan++ got bound with the Burp's Scanner. So, whether it's about auditing the entire application or a specific injection point, the Active Scan++ is always involved within it.

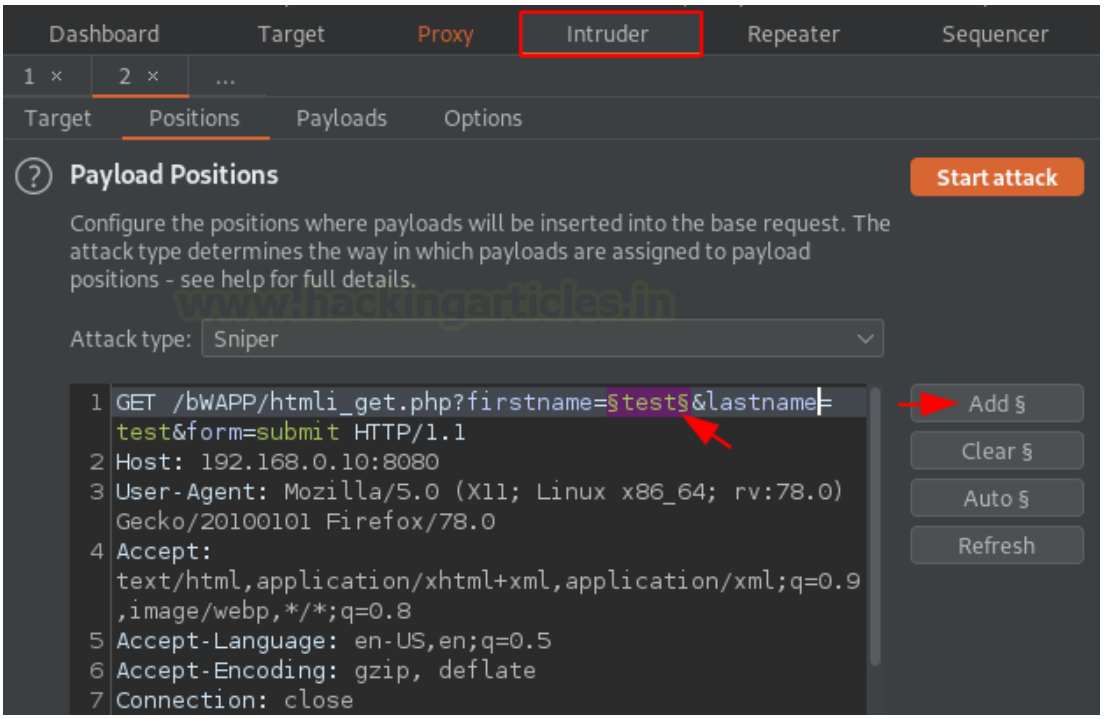
And about the Injection point auditing, so let's do that.

Initiate the bWAPP application with **bee: bug** and then navigate to the **HTML Injection (Reflected)** webpage. Further, we'll enter some test credentials and hit the **Go** button with our **Proxy Service** turned **"ON"**

Over at our burp suite monitor, we got the ongoing HTTP request captured, let's share it with the intruder.



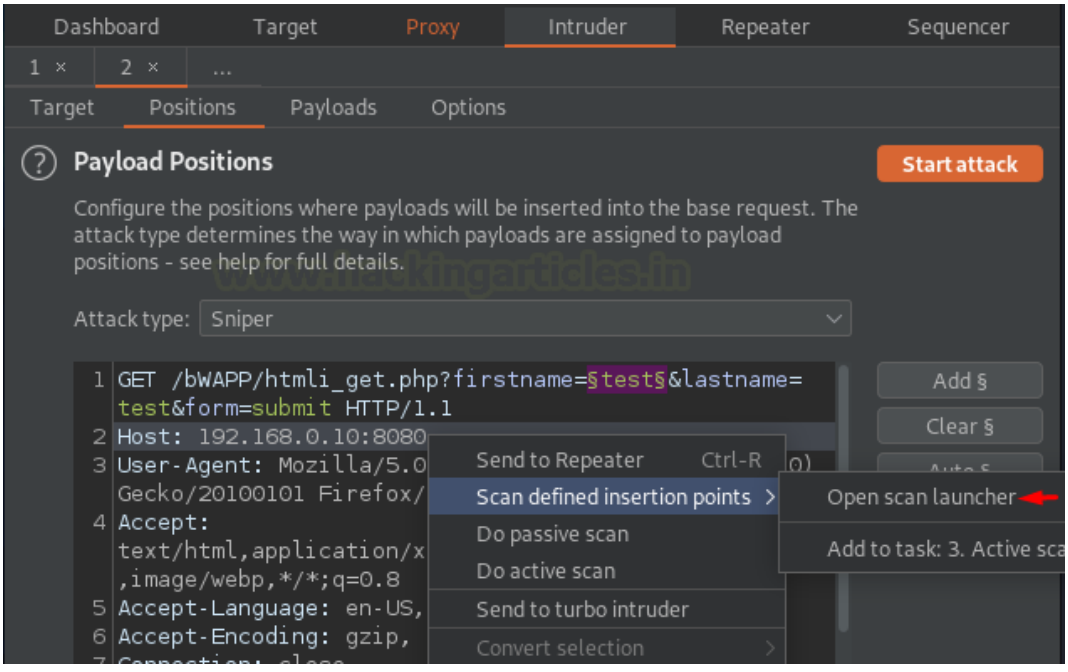
Once done, let's navigate to the **Position section** over at the intruder tab in order to set the injection point. Simply clear the default selections and add the one you want the scanner to audit.



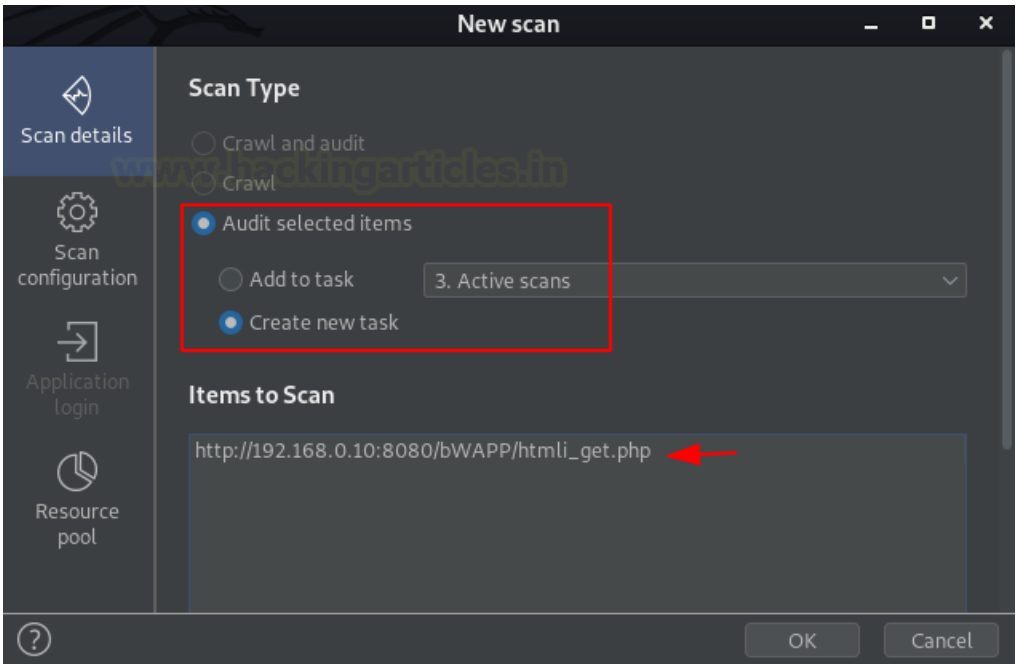
What now? A simple right-click is always our helping hand. So, let's hit the mouse right button and then opt **"Scan defined insertion points".**

As we haven't deleted our previous task, thereby the burp drops two option either to send this request with the ongoing Audit or to start a new one for this.

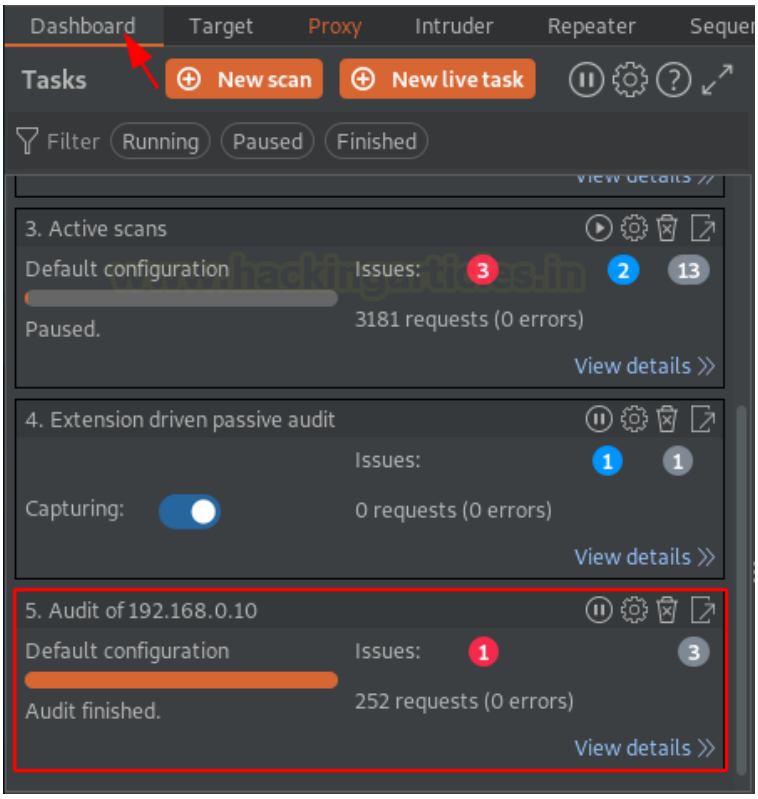
Let's hit the **Open Scan launcher** in order to start and customize the new scan.



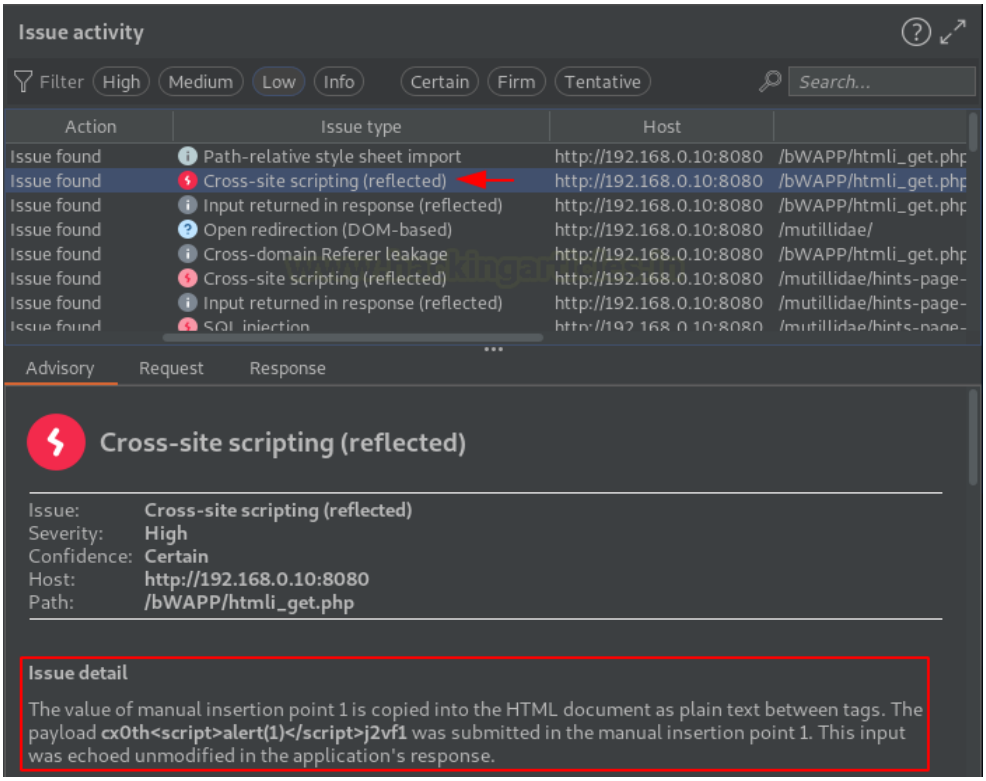
As soon as we do so, we got a window that popped up in front of us stating **"New Scan"**. Let's move forward with the default configuration by hitting the **OK** button.



From the below image we can see that our scanner captured about **3 Basic** and **1 Critical issue** by sharing about 250+ requests at the application's injection point.



Time to analyze. From the below image, we can see that the burp scanner tested Cross-Site Scripting for our injection point and dumped the issue details with the exploiting payload and the mitigation steps.


Issue activity
Filter
High
Medium
Low
Info
Certain
Firm
Tentative
Search...
Action
Issue type
Host
Issue found
Path-relative style sheet import
http://192.168.0.10:8080
/bWAPP/htmli_get.php
Issue found
Cross-site scripting (reflected)
http://192.168.0.10:8080
/bWAPP/htmli_get.php
Issue found
Input returned in response (reflected)
http://192.168.0.10:8080
/bWAPP/htmli_get.php
Issue found
Open redirection (DOM-based)
http://192.168.0.10:8080
/mutillidae/
Issue found
Cross-domain Referer leakage
http://192.168.0.10:8080
/bWAPP/htmli_get.php
Issue found
Cross-site scripting (reflected)
http://192.168.0.10:8080
/mutillidae/hints-page-
Issue found
Input returned in response (reflected)
http://192.168.0.10:8080
/mutillidae/hints-page-
Advisory
Request
Response
Cross-site scripting (reflected)
Issue: Cross-site scripting (reflected)
Severity: High
Confidence: Certain
Host: http://192.168.0.10:8080
Path: /bWAPP/htmli_get.php
Issue detail
The value of manual insertion point 1 is copied into the HTML document as plain text between tags. The payload cx0th<script>alert(1)</script>j2vf1 was submitted in the manual insertion point 1. This input was echoed unmodified in the application's response.

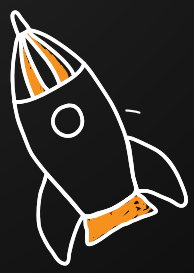
iGNITE Technologies

JOIN OUR TRAINING PROGRAMS

CLICK HERE

BEGINNER

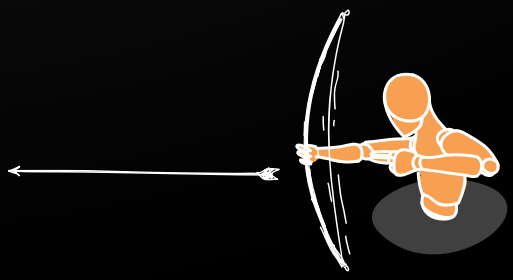
- Ethical Hacking
- Network Pentest
- Bug Bounty
- Wireless Pentest
- Network Security Essentials



ADVANCED

- Burp Suite Pro
- Android Pentest
- Web Services-API
- Advanced Metasploit
- Pro Infrastructure VAPT
- CTF
- Computer Forensics

EXPERT



- Red Team Operation
  - APT's - MITRE Attack Tactics
  - Active Directory Attack
  - MSSQL Security Assessment
- Privilege Escalation
  - Windows
  - Linux