# WEB DEBUGGING BY MODIFYING HTTP TRAFFIC USING FIDDLER

*Computer Network(CSE302) PROJECT REPORT*

*submitted by*

### ANAND RAJ (123004020)

*towards partial fulfillment of the requirements for the award of the degree*

*of*

Bachelor of Technology in

Electronics & Communication Engineering



# School of Electrical & Electronics Engineering

# SASTRA DEEMED TO BE UNIVERSITY

# (A University established under section 3 of the UGC Act ,1956)

# Thirumalaisamudram

# Thanjavur-613401

# 



# School of Electrical & Electronics Engineering

# SASTRA DEEMED TO BE UNIVERSITY

**BONAFIDE CERTIFICATE**

This is to certify that the project work entitled “Web Debugging by modifying HTTP Traffic” is a bonafide record of the work carried out by

# Anand Raj (123004020)

student of Third year B.Tech., Electronics and Communication Engineering, in partial fulfillment of the requirements for the award of the degree of B.Tech. in Electronics & Communication Engineering of the **SASTRA DEEMED TO BE UNIVERSITY, Thirumalaisamudram, Thanjavur - 613401,** during the year 2021-22.

### Name Of The Class Teacher : Prof. PARVARTHY .A SIGNATURE:

Project-Based Work Viva-Voce held on :  **\_** \_\_\_.

### Examiner –I Examiner-II

**ACKNOWLEDGEMENT**

First of all I would express my gratitude to **Prof. Dr S Vaidhyasubramaniam**, Vice Chancellor, SASTRA Deemed University, who provided all facilities and constant encouragement during our study.

I would also like to thank **Dr K. Thenmozhi, Dean (SEEE), Sridhar K, Associate Dean (ECE)** and **R. Chandramouli, Registrar** for their support provided during my course span in SASTRA Deemed University.

I would specially thank and express my gratitude to **PARVARTHY .A, Associate Professor, School of Electrical & Electronics Engineering** for providing me a golden opportunity to do this project and for her guidance and constant support for the completion of this project.

Finally, I thank my parents and my friends for their constant encouragement without which this project would not have been possible.

**ABSTRACT**

Safe browsing techniques help us to get rid out of malicious attacks, phishing or unauthentic sites on Google's list and risky extensions.

 In real life, to establish a connection between the server and to send HTML pages back to the user's web pages or to download data from the server either to the browser or to any requesting application that uses **HTTP**(Hypertext Transfer Protocol)**.**

My project aims to modify and inspect HTTP traffic and to provide explicit caching information. When there is no caching, then chrome has to perform heuristic algorithm to determine whether it responds or not. I used **Fiddler** tool because it allows developers to easily decipher the hidden communications between client and server, allowing for a more holistic view of the Web, its applications, and the delicate interactions that make today’s websites so powerful.

Objective of my project is to speed up our sites and servers by enabling HTTPS debugging option and will also collect traffic captures from end users

using FiddlerCap. HTTP is in the Application layer of the Internet protocol model and in **the Session Layer** of the OSI Model.

So, I concluded we could save number of bytes between two protocol is probably the most important thing in the network scenarios and Fiddler also allows to easily visualize the enhanced UI performance.

# Abbreviations

* **URL :** Uniform Resource Locator
* **IP :** Internet Protocol
* **HTTPS :** Hypertext Transfer Protocol Secure
* **HTTP :** Hypertext Transfer Protocol
* **OSI :** Open Systems Interconnection Model
* **RTT :** Round Time Trip

# TABLE OF CONTENTS

|  |  |  |
| --- | --- | --- |
| **S. No.** | **Title** | **Page No.** |
| 1 | Bonafide Certificate | 2 |
| 2 | Acknowledgement | 3 |
| 3 | Abstract | 4 |
| 4 | Abbreviations | 5 |
| 5 | Table of contents | 6 |
| 6 | List of Figures | 7 |
| 7 | Introduction | 8 |
| 8 | Methodology | 10 |
| 9 | Results and Discussion | 14 |
| 10 | Conclusion | 20 |
| 11 | Future Directions | 21 |
| 12 | References | 22 |

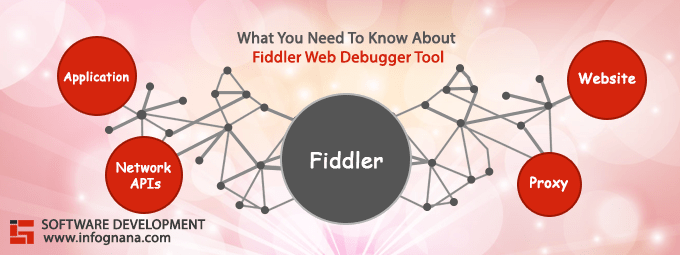
# LIST OF FIGURES

|  |  |  |
| --- | --- | --- |
| **Figure No**. | **Figure Name** | **Page No**. |
| Fig 1.1 | Fiddler's scope | 8 |
| Fig 1.2 | Fiddler interacting with search engine and web server | 9 |
| Fig 2.1 | Fiddler in action viewing Request-Response sequence | 10 |
| Fig 2.2 | Showing Inspector tab | 11 |
| Fig 2.3 | Request Header showing Hexa View | 11 |
| Fig 2.4 | HTTP before comprression | 12 |
| Fig 2.5 | HTTP after comprression | 13 |
| Fig 3.1 | Working with different Protocol | 14 |
| Fig 3.2 | Host, Client and Request group | 15 |
| Fig 3.3 | Breakpoints and Response Status Code | 16 |
| Fig 3.4 | All Web Sessions scenario | 17 |
| Fig 3.5 | Debugged output of applied filters over sessions | 17 |
| Fig 3.6 | Basic request/response statistic | 18 |
| Fig 3.7 | Tested for significant size change in bytes | 19 |

**INTRODUCTION**

In this technological world, an ethical hacker can sniff data by many methods so that developers developed number of Network Snifing tool to trace it out. HTTP Debugger is **a professional http sniffer for intercepting and to analyze the http protocol traffic between a web browser or any application using the HTTP** / HTTPS protocol, and a web server.

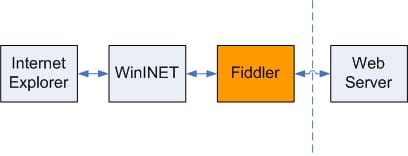
The application that I used is Fiddler to **debug internet packages by using capturing community traffic among the net and take a look at computers**. It enables us to check out incoming and outgoing statistics to reveal and modify requests and responses earlier than the browser receives them.



**Fig(1.1): Fiddler’s scope**

**Tool used : FIDDLER (v5.0.20211)**

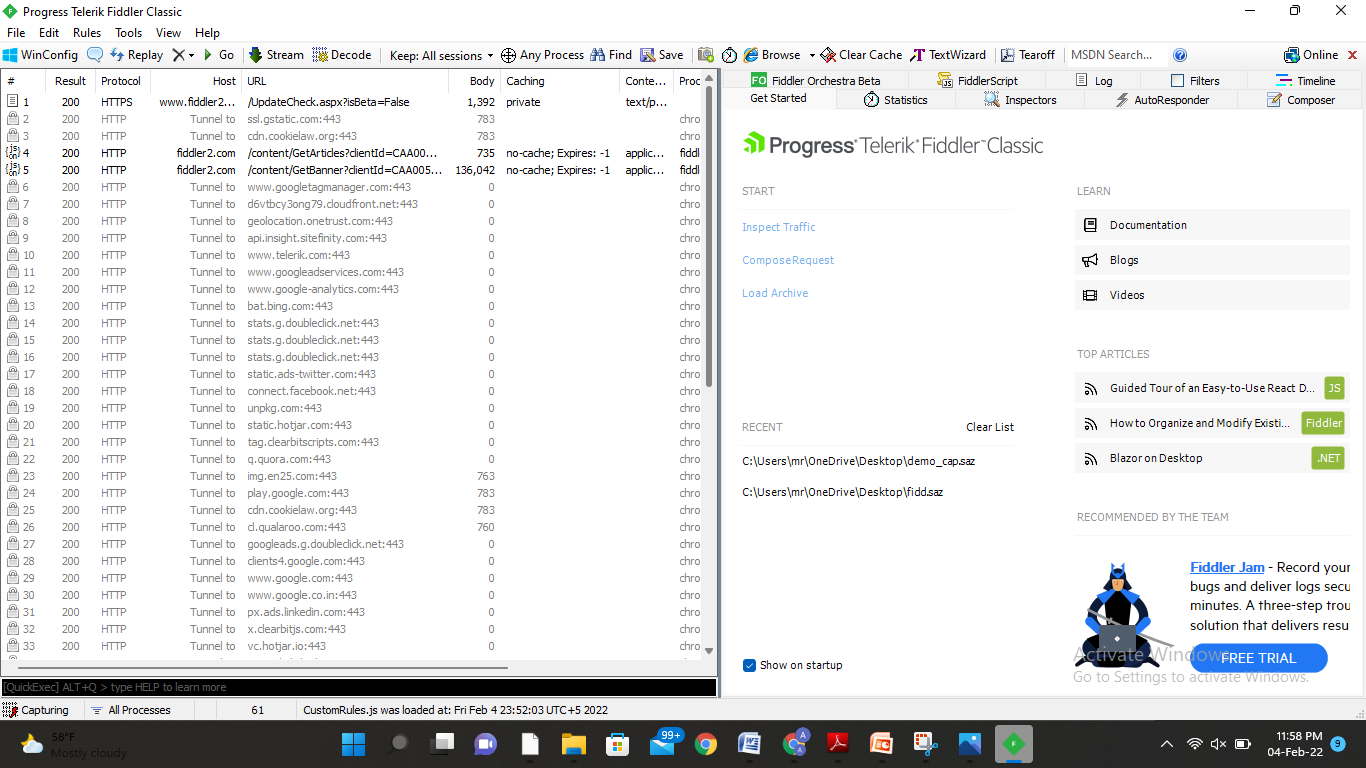
* Tracing tool built specifically for HTTP analysis
* Adjusts browser’s proxy configuration to intercept traffic
* Capture network traffic between the test computers and internet.
* The most powerful web debugging proxy tool for MacOS, Windows and Linux with an enhanced UI.
* Effectively share your findings and add context with built-in team collaboration.
* Customize, save, and share rules to handle complicated requests with ease.
* Unlimited sharing of saved sessions and requests for impressive efficiency gains and receive dedicated email support to stay on track.
* Debugging features are supercharged in Fiddler Everywhere, allowing you to take debugging and testing to the next level without any added effort.

 **Fig(1.2): Fiddler interacting with search engine and Web Server**

# METHODOLOGY

**Viewing and Inspecting Sessions:**

By using default, Fiddler does a model test on startup; for the reason that model take a look at uses HTTP, we can look at the model using Fiddler itself. Similarly, while we open an IE window, we are able to discover that it’s site visitors seems in Fiddler’s Web Sessions list.



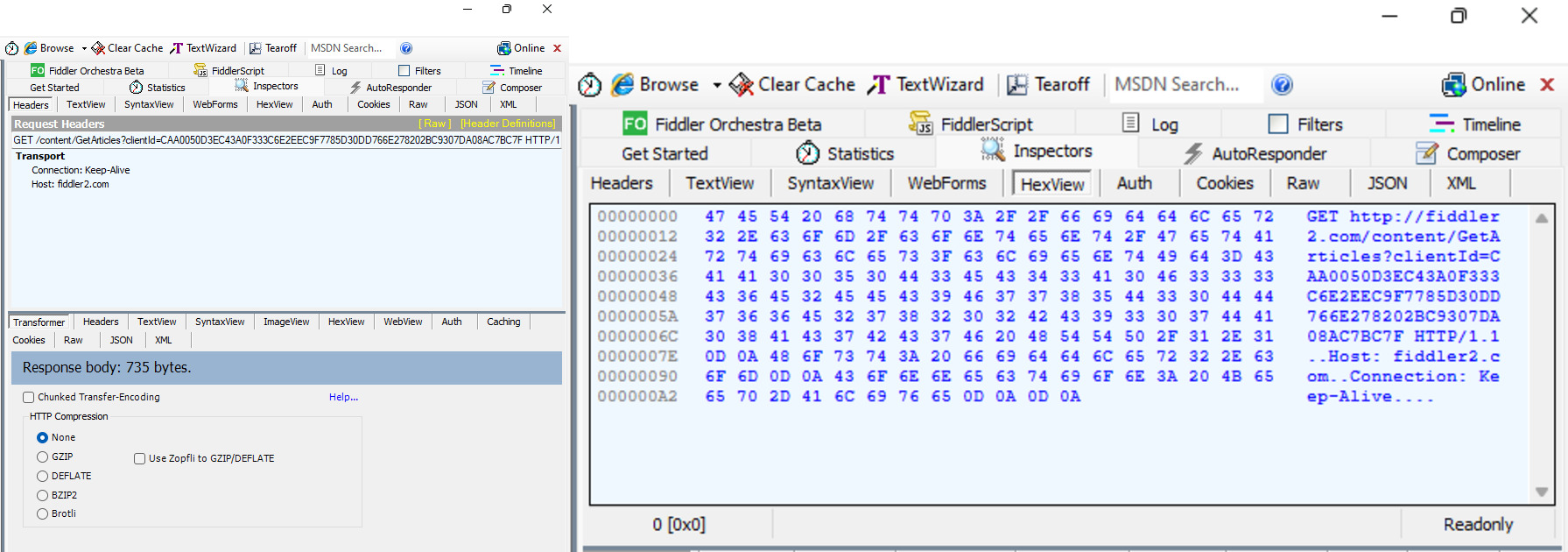
# Fig(2.1): Fiddler in action viewing request-response sequence

Along every session entry, Metadata associated with captured request-response sequences is shown within the session list :-

* **#** :- Incremental ID of the session entry.
* **Result** :- The HTTP response code received.
* **Host** :- In the response header, target host defined.
* **Protocol** :- To transfer session data.
* **URL** :- Target of the request.
* **Body** :- Size of the responses in bytes.
* **Caching** :- Cache settings basically used to determine the response.
* **Process** :- To receive the session data.

**Inspecting Request-Response Sequence:**

Users could inspect and view this data under Inspector tab by selecting a session from session list.



**Fig(2.2): Showing Inspector tab** **Fig(2.3): Request Header showing Hexa View**

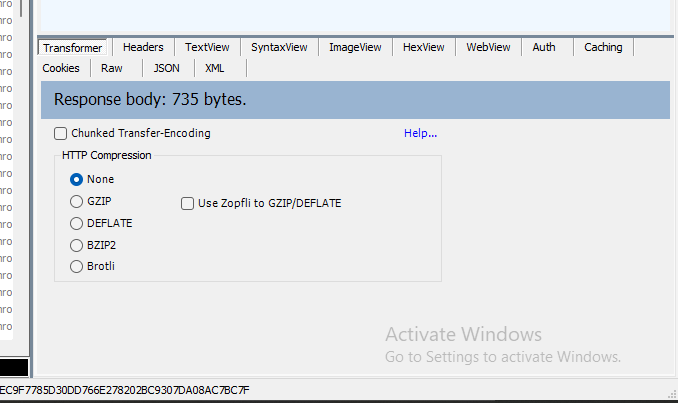
To inspect the data, Headers displays hierarchal tree view of the HTTP request from a client.

If a response is not an image, a developer could request in hexadecimal representation.

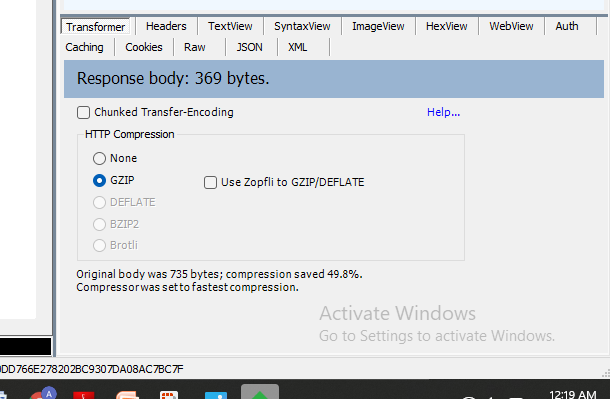
**HTTP Compression:**

During the debugging process, Transformer inspector had plays active role.

This inspector permits the consumer to Compress the HTTP response for simpler and efficient viewing.



# Fig(2.4): HTTP before compression

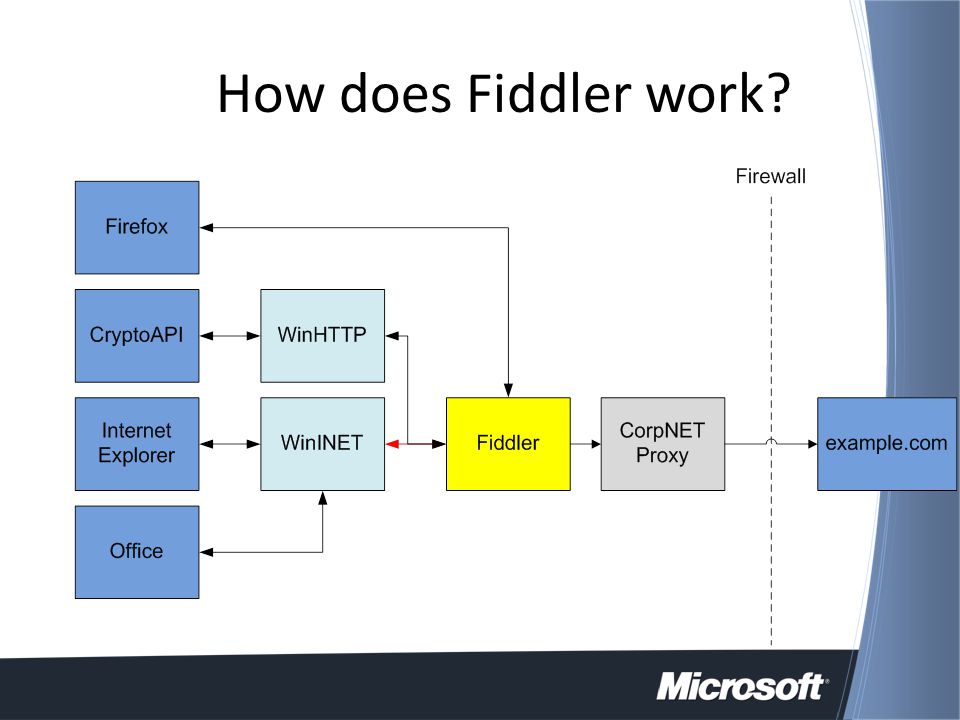


# Fig(2.5): HTTP after compression

The response get compressed using GZIP compression for faster transmission and developers can access gauge impact, such compression may additionally have an overall performance of a web application.

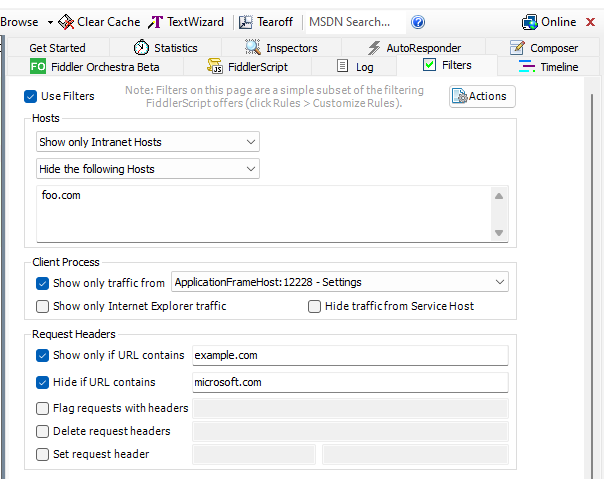
# RESULTS and DISCUSSION

**Debugging and Manipulating Web Sessions using Filters:**

****

**Fig(3.1): Working with different Protocol**

Fiddler offers powerful mechanisms for manipulating a session’s request-response collection. For HTTP debugging, it offers developers with a quick manner to reduce the variety of displayed sessions.

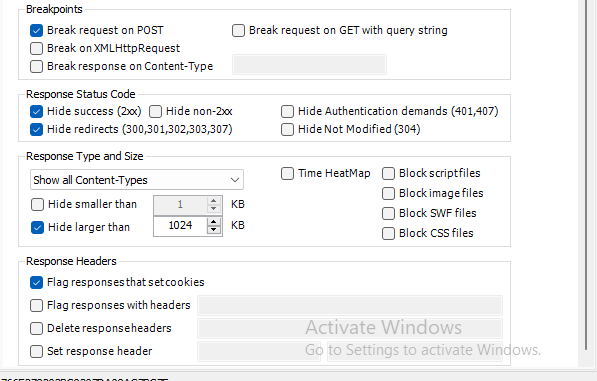


**Fig(3.2): Host, Client and Request group**

We could limit the number of items shown above can specify that options in the selection box. The Hosts group restricts to the show the sessions based on the specific name of the host requested and/or the IE zone.

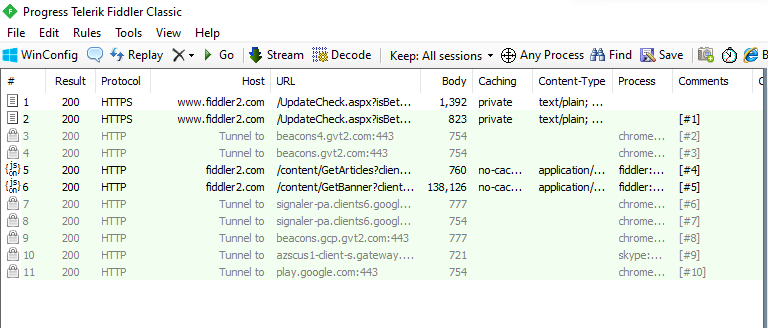
**Settings and Using Breakpoints:**

The Breakpoints permits developer to trigger automatic breakpoints on Requests or Responses meeting not unusual criteria. But as far we saw in different debuggers, breakpoints “pause” the period of execution and allow the consumer to adjust data earlier than execution proceeds.



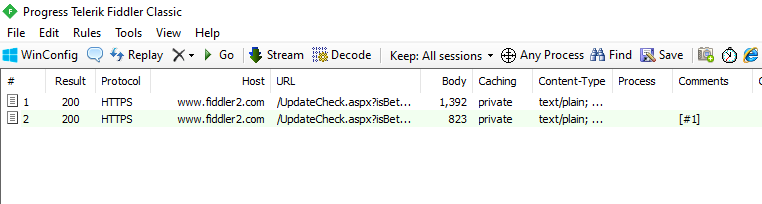
**Fig(3.3): Breakpoints and Response Status Code**

Before applying Filters all web sessions Network Traffic captured, shows below in the table:



**Fig(3.4): All Web Sessions scenarios**

Developers customized the HTTP Network Traffic by selecting appropriate panel from Filter tab.

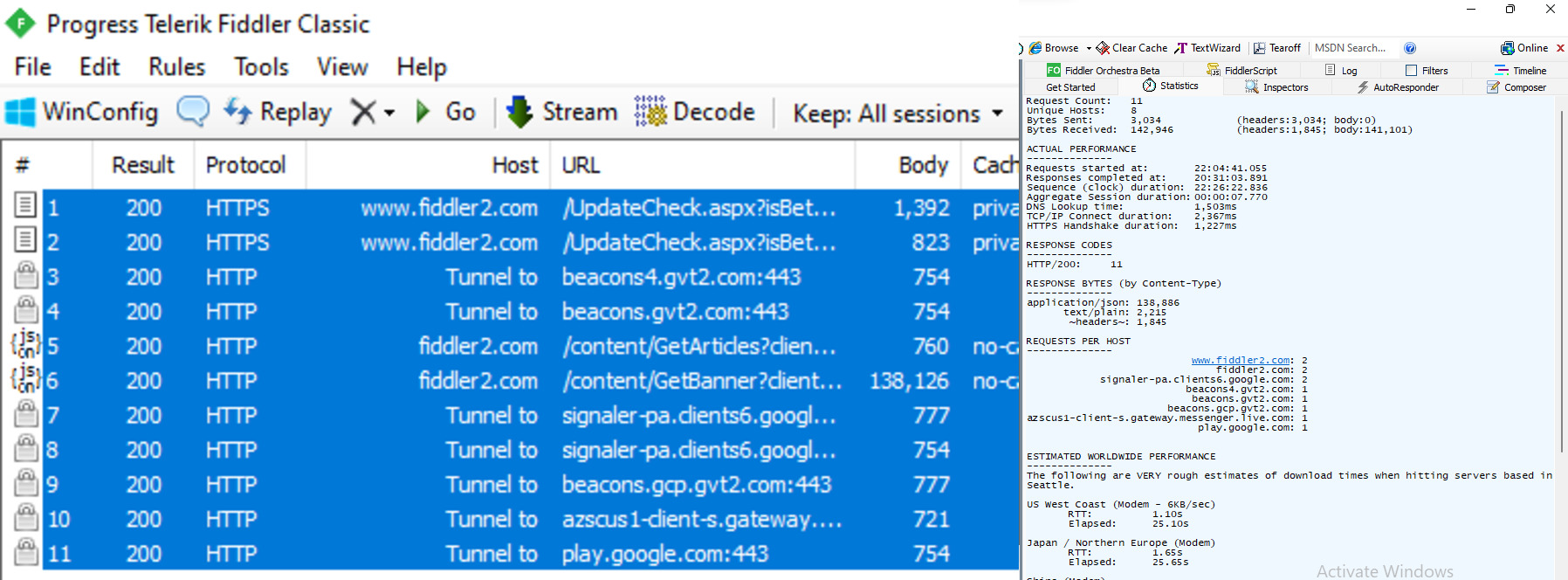


**Fig(3.5)**: **Debugged output of applied filters over Session**

# Analyzing Site Performance

Fiddler provides powerful gear for assessing a page’s potential to carry out different situations, and client profiles. Fiddler provides information to help developers to tune cache execution, allowing them to be sure that their servers are marking each response with the proper caching directives.

We could display the statistics of the complete transaction by selecting all sessions from the session list. The Statistics tab provides approximate estimation about different locales and at different bandwidths.

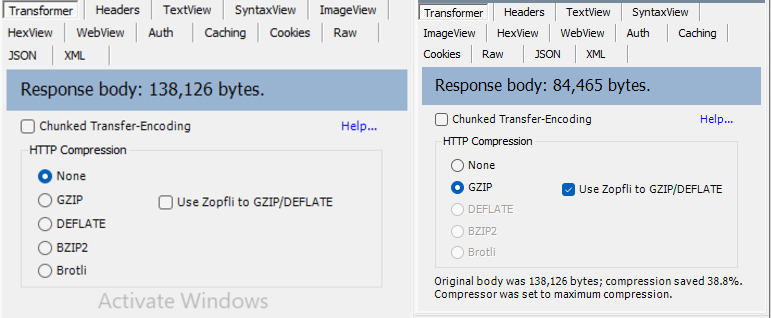
****

**Fig(3.6): Basis request/response statistic**

**Evaluating Cache Performance and Compression Settings:**

Developers should clear both the WinINET cache and stored cookies, before capturing performance data from Fiddler.

Caching assures that the client never requests data uselessly, and it leads to proper cache performance is one of the most important scene of a web site’s overall performance experience.

****

**Fig(3.7): Tested for significant size change**

Original body was 138,126 bytes; compression saved 38.8% compressor was set to maximum compression.

# CONCLUSION

The number of bytes we saved across the wire is probably the most important thing in most network performance scenarios and so using thoughtfully is a great way to improve that Fiddler allow us to easily visualize the performance improvement.

Hence I concluded that:

* By capturing network traffic between internet and test computer, we could customize the useless cache.
* We can adjust browser’s proxy configuration to intercept traffic.
* By tuning cache execution, it leads to proper cache performance is one of the most important scene of a web site’s overall performance experience.

Now a days, modifying HTTP traffic using Fiddler is best debugging proxy tool used to inspect and it is also used to evaluate cache performance for better connectivity and correct output in web pages.

# FUTURE DIRECTIONS

Safe browsing techniques help us to get rid out of malicious attacks, phishing or unauthentic sites on Google's list and risky extensions.

So, Fiddler came with an end-to-end troubleshooting solutions that helps users to support and solve real life problems quickly. Chrome extension of Fiddler can boost remote debugging security credibility with end users while delivering best troubleshooting solutions to provide support for all sizes of team.

It provides astounding functionality aimed towards assisting us to get the reason of problems quickly while debugging. Only Fiddler comes with enhanced UI, that makes the debugging processes better and faster.

Fiddler Everywhere—create rules “out of the box”, mark web sessions, test web adjustments quickly, and make the process of debugging and development less complicated!

# REFERENCES

**Lawrence, E.**, 2012. *Debugging with Fiddler: The complete reference from the creator of the Fiddler Web Debugger*. Eric Lawrence.

**Kilari, V.T., Xue, G.** and **Li, L.**, 2015, December. Host Based Detection of Advanced MiniDuke Style Bots in Smartphones through User Profiling. In *2015 IEEE Global Communications Conference (GLOBECOM)* (pp. 1-6). IEEE.

**Lakshmiraghavan B.** Debugging and Tracing. InPractical ASP. NET Web API 2013 (pp. 27-53). Apress, Berkeley, CA.

**Kirk, A., Rice, S., Azar, Z.** and **Wang, Y**., Private Information Leakage on the Mobile Web.

**Černý, T.** and **Donahoo, M.J.**, A Tool for Evaluation and Optimization of Web Application Performance.