FACTORS AFFECTING THE PERFORMANCE OF WEB APPLICATION FIREWALL

Co-authored by Dainya Thomas-Reynolds

Sergey Butakov

Project report

Submitted to the Faculty of Graduate Studies,

Concordia University of Edmonton

in Partial Fulfillment of the

Requirements for the

Final Research Project for the Degree

MASTER OF INFORMATION SYSTEMS SECURITY MANAGEMENT

Concordia University of Edmonton

FACULTY OF GRADUATE STUDIES

Edmonton, Alberta

December 2020

FACTORS AFFECTING THE PERFORMANCE OF WEB APPLICATION FIREWALL

Dainya Thomas-Reynolds

Approved:

Sergey Butakov [Original Approval on File]

Sergey Butakov Date: December 10, 2020

Primary Supervisor

Edgar Schmidt [Original Approval on File]

Edgar Schmidt, DSocSci Date: December 14, 2020

Dean, Faculty of Graduate Studies

Factors Affecting the Performance of Web Application Firewall

Dainya Thomas-Reynolds
Concordia University of Edmonton
Edmonton, Alberta, Canada

Sergey Butakov
Concordia University of Edmonton
Edmonton, Alberta, Canada

ABSTRACT

Web application firewalls (WAF) have been proven to be a useful tool in a much needed security for internet applications. Automation of ruleset development for WAF may reduce human errors and establish required baseline for web security. This research paper shows that mechanical implementation of highest paranoia levels on WAF may severely affect the productivity of the system and calls for the need for human intervention to critically review the proposed ruleset and find the optimal paranoia level that suits specific web application.

Keywords: firewall automation; network security; web application firewalls; paranoia levels, ruleset.

INTRODUCTION

Web-based applications have completely revolutionized the face of communication. The readiness, the increased online-based businesses and services, the increased need for information, and the reachability of web services have collectively increased efficiency and operational productivities in society. However, with these increases, there is a call for a greater need of information protection. Use of web applications has enabled a completely new environment of vulnerabilities and exploits such as SQL injection, Cross-site scripting (XSS) and many more.

The escalation in breaches and the complexity of attacks and services necessitates supplementary functionality beyond the capability of dated network-based security products. The

integration of web application security offers a complete solution to help decrease threats associated with the security of web-based services

This paper will review the fundamental functionality of ModSecurity and the factors that affect it's performance such as Paranoia Levels (PL). Web Application Firewalls (WAFs) became a vital staple as a layer of protection for online systems from nefarious actors. The effectiveness of a firewall is linked to the firewall's response time and its performance while under duress. ModSecurity, a WAF is an open source web application used for monitoring, logging and granting or denying access to web-based application. WAF setup and maintenance requires intensive human intervention which makes it prone to human errors. There is an ongoing effort from the information security community to automate WAF configuration to ensure security baseline while reducing negative effects of the human factors.

The goal of this research is to examine the behavior of ModSecurity configured with the latest Core Rule Set (CRS) v3.2 recommended by OWASP (Folini, 2019) to determine the firewall's performance measured by its effectiveness and throughput. Two hypotheses (H1 and H2) will be tested in this research. H1 assumes that PL affects the detection ratio and performance of WAF. H2 assumes that ModSecurity is still useful while configured at the maximum paranoia level (PL4).

LITERATURE REVIEW

Over the last two decades significant number of research projects looked at Web Application Firewalls and how to configure them to guard systems against nefarious actors. One of the goals of this effort from the InfoSec community is to reduce the human factor and have universal approach to establish web application security baseline. According to two web articles, (OWASP top 10 web app, 2019 and Vulnerabilities, 2020), it is clear, that even with so many

efforts, such attacks as SQLi remain the leading threat in web based attacks. The related papers in the literature review showcase numerous ways and techniques that can be deployed to combat web based attacks, still, OWASP Top Ten vulnerability list remains the same, yearly and Core Rule Set (CRS) version keeps getting updated to better fine-tune ModSecurity and other firewall appliances that rely on those CRS. Subsequently, there is still a need for more work to thoroughly test ModSecurity's effectiveness and throughput with recommended security baselines.

Ali et al., carried out an experimental analysis on WAFs to detect Cross-Site Scripting and Injection attack (Ali et al.). The research analyzed the behavior of four WAF namely; ModSec, Sophos, AQTRONIX, and Barracuda to determine detection capabilities. Although all products were able to sustain some attacks, the researchers were able to inject several payloads using various scanners without customizing the attacks, yet, the firewall did not trigger an alarm. These experiments show that out of box vendor solutions needs fine tuning by experienced administrators and thus rely on human expertise that may not always robust and readily available. While Ali et al., focused on analyzing how WAF can detect cross-site scripting (XSS) and injection attacks, Tekerek et al., conducted an experimental development analysis of a Hybrid Web Application Firewall (Tekerek et al. 2014). After the developments of a hybrid approach, it was observed that while signature based detection works faster in detecting certain attacks, it is useless against zero-day attacks. It was concluded that by combining both Anomaly-Based Detection and Signature-Based Detection, they would be more effective in prevention against SQLi and zero-day attacks.

Jagraj et al. conducted research to determine the impact paranoia levels have on the effectiveness of ModSecurity WAF (Jagraj et al. 2018). Tests were performed on ModSecurity

with CRS v3.0 to analyze how the firewall will behave at all four paranoia levels. At PL1, higher number of attacks were able to bypass the firewall. Additionally, PLs were increased to determine the number of false positives that would be logged, and it was observed that legitimate users were not able to register on the website at the maximum PL4 configuration. The study was indicative that PL, if improperly implemented, my severely affect the usability of the web application. The experiment done by Oberoi et al. started out by configuring ModSecurity with and without CRS, then by running multiple tests, were able to determine the change in response time of the firewall. According to the obtained results, the increase in CRS drastically decreased the firewall's performance in serving requests (Oberoi, Samuel and Zarvarsky 2018).

The essence of the proposed research is to explore questions that have been left open by research papers reviewed above. Even though Ali et al. (Ali et al.) looked at multiple firewalls to determine each firewall's performance in detecting payloads of Cross-site scripting and Injection attacks and (Tekerek et al. 2014) (Jagraj et al. 2018) (Oberoi, Samuel, and Zarvarsk 2018) all focused on other techniques to detect, block and log SQLi attacks, the researchers did not analyze how the paranoia level affects the firewall performance in terms of detection efficiency and negative impact on application usability. There are still areas that requires development for the effects of paranoia levels on throughput and effectiveness and with that, this research aims to analyze the behavior of ModSecurity latest CRS to determine what rules are being triggered at each of the four PL(1-4) and how the firewall performs at each level while under duress by the performance tool, Siege. It is important to note that web application security appliances will never replace security professionals. Out of the box, ModSecurity CRS is configured at PL1 and therefore, the firewall is performing below average. For the firewall to be efficient and effective,

both software and human intervention is required to fine tune the application to meet organization's business objectives, requirement and goals.

The performance of the firewall will be analyzed using a popular automation benchmark tool known as Siege. Knowing how the firewall performs at each of the PL, will provide guidance for security professionals responsible for reviewing and updating CRS, to adjust how traffic is monitored and how rules are triggered, which will improve how quickly attacks are blocked and logged, improve data transmission to legitimate users and how responsive the web application is while multiple users access the web services.

EXPERIMENTAL STUDY OF WAF AND PARANOIA LEVELS

In this section, the focus will be on devising the methodology structure and answering questions on how PL affects the performance of the firewall with the help of experimental results obtained from this research. The steps involved in carrying out the experiment is delineated below:

Methodology and Lab Description

The two hypotheses that will be tested in this research are as follows:

H1. PL affects the detection ratio and performance of WAF.

H2. ModSecurity is still useful while configured at the maximum level, PL4.

The following steps have been done to test both hypotheses: (1) Setup testbed conducive for network monitoring and attack simulation. (2) ModSecurity configured to detect usage on the web application. To determine its effectiveness and throughput a benchmark performance tool known as Siege, will be utilized. ModSecurity will be configured with PLs ranging from PL1 to PL4 and its performance will also be analyzed using Siege.

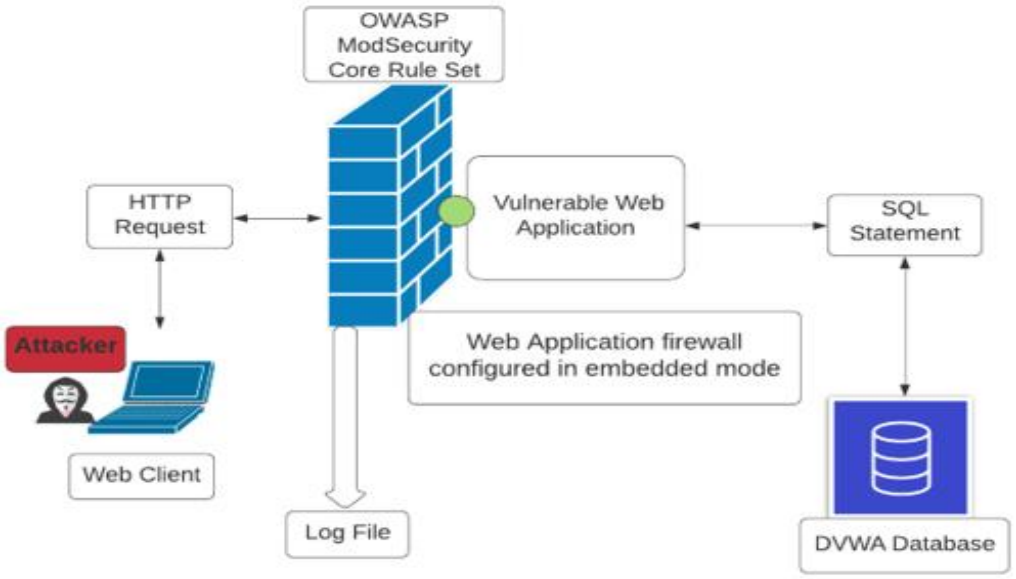


Figure 1. Network design setup for conducting experiments for firewall performance analysis

The network diagram on Figure 1 depicts a simulation of the setup where the attacker or user(s) sends HTTP requests to the DVWA. The DVWA (EthicalHacker, 2020), is setup to act as the test application having vulnerable codes that can be exploited using several types of attacks including SQL injection and cross-site scripting. ModSecurity, which is acting as the firewall, is configured in Embedded mode for the web service. Configuring it as such limits new points of failure and scales seamlessly as the underlying web infrastructure scales. The firewall is configured at varying Paranoia Levels (1-4) and will either block or allow the traffic to pass to the DVWA database and each action is logged in the firewall's logfile. The performance tool, Siege will be utilized to analyze the firewalls behavior and performance while the firewall is configured at the vary paranoia levels (PL1-PL4).

Experimental Results and Analysis

To test ModSecurity, a performance tool known as Siege was used to give an overview of how the firewall performs under duress. The tool works by simulating hits to the web server with

concurrent connections from virtual users. Simulation for each paranoia level was done with 10, 20, 30, 40 and 50 concurrent users. It can be observed from the results that for PL2 to PL4 the application server's availability was affected by the change in the paranoia level setting, while at PL1 the server's availability remained at full capacity. Availability is calculated by the percentage of socket connections successfully handled by the server. It can be seen from tables 3 and 4 that under PL3 and PL 4 servers availability went below 98% which is not acceptable for a typical e-commerce application. Another critical indicator – response time went up to almost 2 seconds with PL4 under a mild workload of 10 concurrent users and went to unacceptable time of 8.20 seconds for highly loaded situations with PL4. Although the poor response time could be addressed by vertical or horizontal scaling of the server setup, the same scaling would be required for WAF as well, thus significantly increasing demand for resources.

The results also indicate that concurrency increases as the server's performance decreases. Concurrency can be calculated by the average of all simulated users divided by the elapse time. Table 1 with PL1 has the least amount of concurrent value of 11.3 at a workload of 10 concurrent users and PL4 shows a significant increase of 22.1 under the same workload of 10 concurrent users. Such results are deemed unacceptable for a typical e-commerce application because at PL4, 50 concurrent users are all it takes to render the system ineffective.

Hypothesis H1 was clearly proven by the decrease of the performance for the same workload under the same PL. Reduction in availability and throughput as well as increase of elapse and response times between PL1 and PL4 are visible in the Appendix 1.

H2 looked at the usability issues as time is a precious commodity and response time delays in using web services and applications take users away from a service (Dennis and Taylor 2005). According to table 4 in Appendix A, PL4 was configured on the firewall and the

performance was drastically impacted, even at the lowest concurrency level with 10 users, c10. The higher the PL, the more rules become active which means all HTTP packets are checked against each criteria/ rule that are defined, placing enormous pressure on the firewall, thus resulting in delays, dropped packets or even false positive. In the paper published by ((Dennis and Taylor 2005) reasonable response time is 2 s and is considered "short", while 10 s is considered "long". " and if the application is not vital to users' needs, it may be seen marginal in terms of acceptability. Another workload indicator – Availability dropped from 100% to 88% which is not acceptable for generic e-commerce applications. Although the availability issue can be addressed with the horizontal or vertical scaling of the infrastructure, under the current testbed settings H2 needs to be rejected. A paper published by (Sobola, Zavarsky and Butakov, 2020) mentioned the use of a use case by Trustwave Holdings Inc (SpiderLabs, 2020) where it introduced strict profile checks using postive or negative security model. However, (Sobola, Zavarsky and Butakov, 2020) recommended bridging the gap between positive and negative security models that may improve performance for the web application. This approach could be an interesting research to be undertaken in the future.

CONCLUSION

Conducted experiment indicated that basic introduction of the rulesets configured as per highest paranoia level can severely affect the performance of the web application. With each paranoia level setting that was tested, different rulesets were activated. Each HTTP packet goes through rigorous scans to ensure it should be allowed or be denied and the more rules that are enabled at each paranoia level, the harder the firewall must perform to scan each HTTP packet. With that, the future of this research would be to analyze the rulesets for each of the different paranoia levels that were activated, and provide this data to administrators who are responsible

for writing and testing those rulesets to review and make changes where necessary to improve the performance of ModSecurity, therefore, a semi-supervised simulation of the ruleset would be idealistic.

Additionally, the rulesets were written by humans, therefore, it can be assumed that human-errors were introduced in the rulesets which may have resulted in the experimental outcome as outlined in the appendix section of this research, therefore, the future direction of this research may seek to investigate whether there are certain biases or error that people may have introduced in each paranoia level rulesets.

This research is the first step in the study that aims to finding a solution to achieve a balance between standardization of the rulesets in web application firewalls and human intervention required to finetune rulesets for specific applications.

REFERENCES

Ali, R., Zavarsky, P., & Lindskog, D. (n.d.). Experimental Analysis Web Application Firewalls to Detect XSS and Injection Attacks. Retrieved 05 27, 2020

Dennis, A., & Taylor, N. (2005, 05 30). Information foraging on the web: The effects of acceptable Internet delays on multi-page information search behavior. Retrieved 06 23, 2020

EthicalHacker. (2020). *Github*. Retrieved 05 27, 2020, from https://github.com/ethicalhack3r/DVWA

Folini, C. (2019, 07 22). *Including OWASP ModSecurity Core Rule Set*. Retrieved from https://www.netnea.com/: https://www.netnea.com/cms/apache-tutorial-7_including-modsecurity-core-rules/

Jagraj Singh, J., Hamman, S., & Zavarsky, P. (2018). Impact of Paranoia Levels on the Effectiveness of the ModSecurity Web Application Firewall. *2018 1st International Conference on Data Intelligence and Security (ICDIS)*. South Padre Island, TX, USA. Retrieved 05 28, 2020, from https://ieeexplore.ieee.org/abstract/document/8367754?casa_token=XdNz-nrwz9kAAAAA:-P8-0cCEvRClEe8MDVhhu0LjBfg9vh-NIioTFRjDNxoLluPiUlQrrdlFw57qHO05_oMUwMez

Oberoi, J., Samuel, H., & Zarvarsk, P. (2018). How much web security is just enough? Analysis of Granulated Web Application Firewall Rules on Web Server Performance. *17th IEEE International Conference On Trust Security And Privacy In Computing And Communications.* New York, USA.

OWASP top 10 web app vulnerabilities over time, n.d (2019, April 5). Retrieved from https://medium.com: https://medium.com/@TechExpertise/owasp-top-10-web-app-vulnerabilities-over-time-8997af9cb13a

Sobola, T., Zavarsky, D., & Butakov, S. (2020). EXPERIMENTAL STUDY OF MODSECURITY WEB APPLICATION FIREWALLS. *2020 IEEE 6th Intl Conference on Big Dara Security on Cloud (BigDataSecurity), IEEE Intl Conference on High Performacne and Smart Computing,(HPSC) and IEEE Intl Conference on Intelligent Data and Security (IDS)*. IEEE.

SpiderLabs, T. H. (2020, Aug 7). *ModSecurity Reference Manual*. Retrieved from SpiderLabs: https://github.com/SpiderLabs/ModSecurity/wiki/Reference-Manual-(v2.x)#Attack_Prevention_and_Virtual_Patching

Tekerek, A., Gemci, C., & Faruk Bay, O. (2014). Development of a Hybrid Web Application Firewall to Prevent Web Based Attacks. *2014 IEEE 8th International Conference on Application of Information and Communication Technologies (AICT)*. Astana, Kazakhstan. Retrieved 05 27, 2020

Vulnerabilities Over Time, n.d (2020). Retrieved from https://sucuri.net: https://sucuri.net/guides/owasp-top-10-security-vulnerabilities-2020/

APPENDIX A:
PERFORMANCE RESULTS OF MODSECURITY USING SIEGE PERFORMANCE TOOL WHILE FIREWALL IS SET AT PL1 to PL**4.** ***C= Concurrent Users***

Table 1. Firewall performance at PL1

Performance characteristics	**C10**	**C20**	**C30**	**C40**	**C50**
Transaction	800	1600	2400	3200	4000
Availability	100%	100%	100%	100%	100%
Elapsed time (sec)	1.03	2.09	3.19	4.29	5.50
Data transferred (mb)	138.54	127.89	122.84	105.75	94.82
Response time(sec)	0.01	0.01	0.04	0.05	0.31
Transaction rate (Tans/sec)	776.7	765.6	752.4	745.9	725.3
Throughput (mb/sec)	2.67	2.42	2.30	2.18	2.08
Concurrency	11.3	16.9	20.6	24.5	29.1

Table 2. Firewall performance at PL2

Performance characteristics	**C10**	**C20**	**C30**	**C40**	**C50**
Transaction	800	1600	2400	3200	4000
Availability	100%	100%	99.94%	99.54%	99.12%
Elapsed time (sec)	1.11	2.23	3.44	4.89	6.52
Data transferred (mb)	120.84	117.59	103.75	90.98	87.62
Response time(sec)	0.09	0.10	0.12	1.95	2.03
Transaction rate (Tans/sec)	720.7	717.5	697.7	654.4	613.5
Throughput (mb/sec)	2.45	2.21	2.12	1.87	1.47
Concurrency	15.1	18.3	22.7	27.8	33.9

Table 3. Firewall performance at PL3

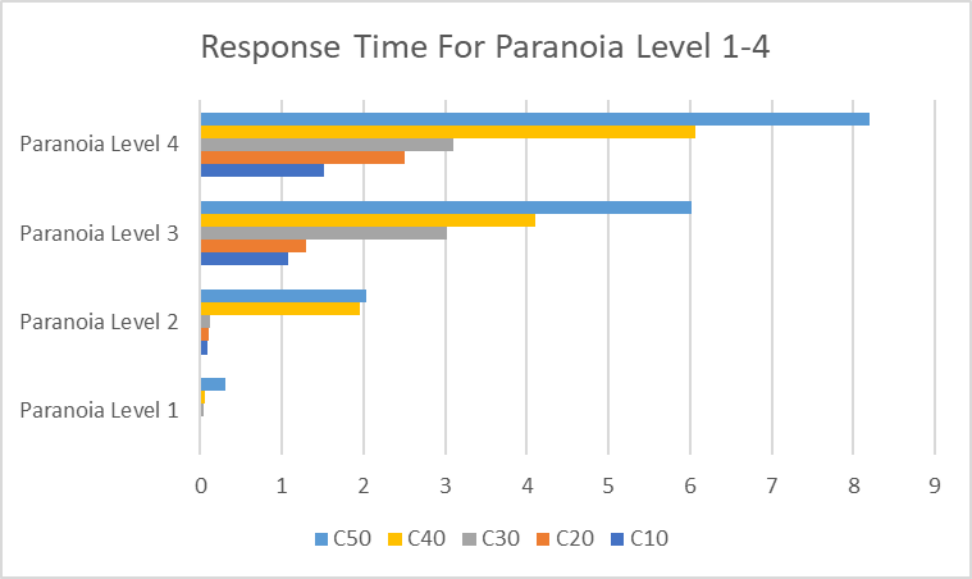
Performance characteristics	**C10**	**C20**	**C30**	**C40**	**C50**
Transaction	800	1600	2400	3200	4000
Availability	99%	98.45%	98.81%	98.27%	98%
Elapsed time (sec)	1.35	2.75	5.23	7.3	9.42
Data transferred (mb)	117.74	113.79	91.76	82.59	74.26
Response time(sec)	1.08	1.30	3.02	4.10	6.02
Transaction rate (Tans/sec)	592.6	581.8	458.9	438.4	424.6
Throughput (mb/sec)	2.03	1.84	1.31	1.13	1.06
Concurrency	20.9	25.5	32.50	36.9	40.3

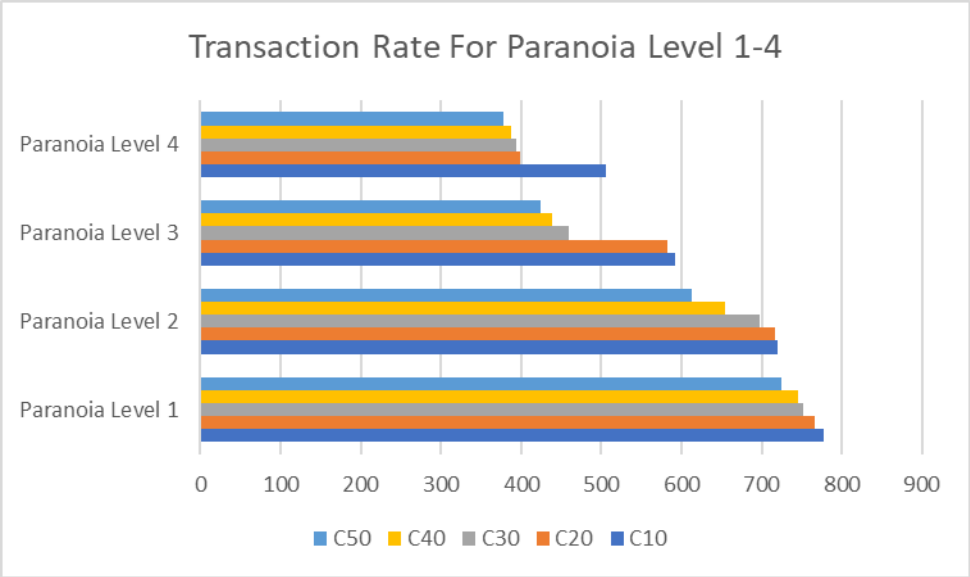
Table 4: Firewall performance at PL4.

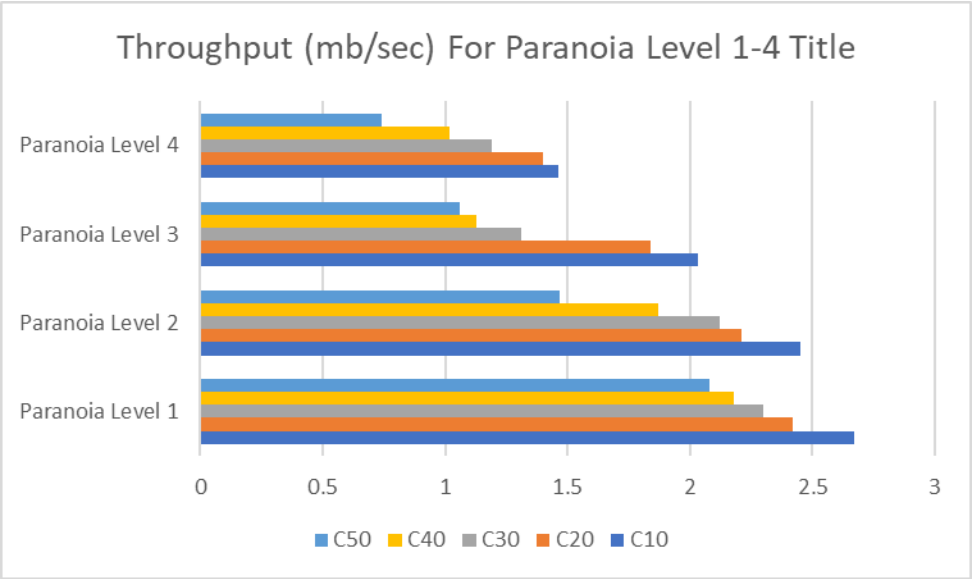
Performance characteristics	**C10**	**C20**	**C30**	**C40**	**C50**
Transaction	800	1600	2400	3200	4000
Availability	93.10%	92%	89.11%	88.36%	88%
Elapsed time (sec)	1.58	4.01	6.08	8.25	10.60
Data transferred (mb)	114.94	106.89	86.82	71.72	68.74
Response time(sec)	1.51	2.50	3.10	6.06	8.20
Transaction rate (Tans/sec)	506.3	399	394.7	387.8	377.4

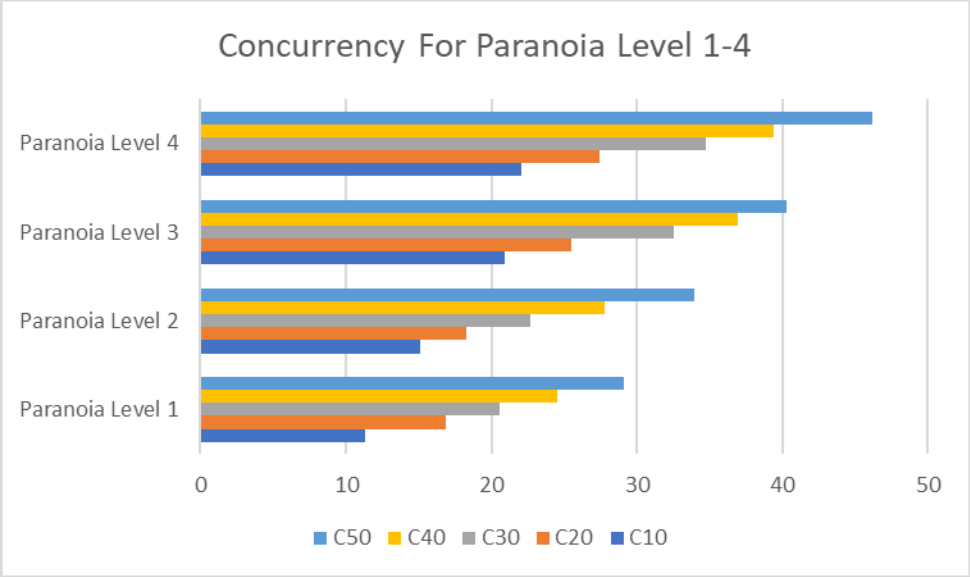
Throughput (mb/sec)	1.46	1.40	1.19	1.02	0.74
Concurrency	22.1	27.4	34.7	39.4	46.2

APPENDIX B:
GRAPH OF PERFORMANCE RESULTS OF MODSECURITY USING SIEGE PERFORMANCE TOOL WHILE FIREWALL IS SET AT PL1 to PL4






Throughput (mb/sec) For Paranoia Level 1-4 Title
Paranoia Level 4
Paranoia Level 3
Paranoia Level 2
Paranoia Level 1
0
0.5
1
1.5
2
2.5
3
C50 C40 C30 C20 C10


Concurrency For Paranoia Level 1-4
Paranoia Level 4
Paranoia Level 3
Paranoia Level 2
Paranoia Level 1
0
10
20
30
40
50
C50 C40 C30 C20 C10