BERT-Based RASP: Enhancing Runtime Application Security with Fine-Tuned BERT[1]

Mayur Sinha
sinhamayur@yahoo.com

Sangram Kesari Ray
shankar.ray030@gmail.com

Khirawadhi
Khirawdhi@gmail.com

January 3, 2024

Abstract

Runtime Application Security Protection (RASP) is crucial in safeguarding applications against evolving cyber threats[2]. This research presents a novel approach leveraging a fine-tuned BERT (Bidirectional Encoder Representations from Transformers)[3] model as the cornerstone of a robust RASP solution. The fine-tuning process optimizes BERT's natural language processing[4] capabilities for application security, enabling nuanced threat detection and mitigation at runtime. The developed RASP system harnesses BERT's contextual understanding to proactively identify and neutralize potential vulnerabilities and attacks within diverse application environments. Through comprehensive evaluation and experimentation, this study demonstrates the efficacy and adaptability of the BERT-based RASP solution in enhancing application security, thereby contributing to the advancement of proactive defense mechanisms against modern cyber threats.

1 Introduction

Traditional RASP tools perform anomaly[5] detection using techniques often rely on pattern matching and statistical analysis, often falling short in capturing complex, context-dependent anomalies prevalent in network traffic. The focus of this study centers on the augmentation of Runtime Application Security Protection (RASP) mechanisms through the integration of a fine-tuned BERT model trained on the CSIC dataset. By infusing BERT's contextual understanding and nuanced comprehension of payload semantics, this research endeavors to enhance the detection and mitigation of threats[6] lurking within application payloads during runtime.

2 Methodology

2.1 Dataset for evaluation

A validation-set[7] based on csic-dataset[8] has been used for evaluating the RASP solution.

2.2 Development Setup

We've created a generic POST-endpoint using FAST API[9], upon which we'll fire the payloads using an HTTP-client that iterates through the validation-set.

We're using a FAST API global dependency hook to inspect the request hitting the POST-endpoint. The global dependency hook extracts the payload from the request, pre-processes it to convert the payload to a format suitable to the Fine-Tuned BERT for inference. The Fine-Tuned BERT outputs the probabilities for the payload as Normal and Anomalous.

We throw a custom HTTPException if payload has high probability as Anomalous.

2.3 Software and Tools

We've used FAST API for developing the POST-endpoint. We've used transformers, and datasets library from huggingface for loading the fine-tuned model and our dataset for evaluation. We're using CPUs for inference.

2.4 Evaluation Metrics and Definitions

Accuracy is used to assess the overall performance of the fine-tuned BERT model.

$$\text{Accuracy} = \frac{\text{TP} + \text{TN}}{\text{TP} + \text{TN} + \text{FP} + \text{FN}}$$

Accuracy: Accuracy measures the overall correctness of the model's predictions.

True Positive (TP): The number of instances correctly classified as positive by the model.

False Positive (FP): The number of instances incorrectly classified as positive by the model when they are actually negative.

True Negative (TN): The number of instances correctly classified as negative by the model.

False Negative (FN): The number of instances incorrectly classified as negative by the model when they are actually positive

3 Results

3.1 Performance Metrics

Our RASP setup has reported 75% accuracy with the following confusion-matrix.

	Actual Positive	Actual Negative
Predicted Positive	2927	1588
Predicted Negative	195	2467

3.2 Discussion of Findings

The implementation of a fine-tuned BERT model in Runtime Application Security Protection (RASP) shows promising results, achieving a 75% accuracy rate in identifying cyber threats. This integration signifies a substantial advancement in cybersecurity, leveraging the capabilities of machine learning to enhance traditional RASP tools. However, the model's limitations, such as the potential for false positives and false negatives, must be acknowledged. It is crucial to consider ways to further improve accuracy, possibly by integrating additional data sources or using more advanced training techniques. The practical implications of these findings are significant, suggesting that incorporating advanced AI models into cybersecurity tools can lead to more proactive and effective defenses against evolving cyber threats. This research sets a foundation for further exploration in the field, aiming to develop more robust and intelligent security solutions.

3.3 Significance and Implications

The integration of BERT in RASP has significant implications for cybersecurity. It demonstrates a shift towards AI-enhanced security systems capable of adapting to new threats. This approach could reshape the landscape of cybersecurity, offering more dynamic and responsive protection mechanisms. Furthermore, it highlights the potential of machine learning in threat detection and response, encouraging further research and investment in this area. The findings suggest that such integrations can greatly benefit organizations in protecting their digital assets, potentially setting new standards in cybersecurity practices. This research could serve as a catalyst for more AI-driven security solutions, ultimately contributing to a safer digital environment.

4 Data Availability

The code can be downloaded from here: https://github.com/b1nch3f/bert-rasp

References

[1] https://huggingface.co/b1nch3f/owasp-bert.

[2] Owasp TOP10. Web application security risk. https://owasp.org/www-project-top-ten/. 2021.

[3] J. Devlin, M.-W. Chang, K. Lee, and K. Toutanova. Bert: Pre-training of deep bidirectional transformers for language understanding. 2018.

[4] D. Jurafsky and J. H. Martin. Speech and language processing. 2020.

[5] M. E. H. Holm. Estimates on the effectiveness of web application firewalls against targeted attacks pp. 250–265. 2013.

[6] WASC. Web application security consortium. http://www.webappsec.org/. 2010.

[7] https://huggingface.co/datasets/b1nch3f/csic-dataset.

[8] C. T. Giménez, A. P. Villegas, and G. Á. Marañón. Http dataset csic. 2010.

[9] https://fastapi.tiangolo.com/.