Cloudflare WAF AI

DESIGN DOCUMENT

SDMAY21-16
Client: Cylosoft
Advisors: Dr. Yong Guan, Dr. Berk Gulmezoglu

Ben Feddersen: Stakeholder Correspondent
Eric Reuss: Documentation Manager
Giovanni Mejia: Report Manager
Jordan Heim: Documentation Manager
Presiian Iskrenov:
Ryan Burgett: Meeting Leader

sdmay21-16@iastate.edu
http://sdmay21-16.sd.ece.iastate.edu

Revised: October 4, 2020//V1

Executive Summary

Development Standards & Practices Used

- Agile
- REST
- TCP/IP
- HTTP

Summary of Requirements

Microsoft ISS will generate text-based log files of each web request. A console-based application will be created to find the current log file for the site that is being monitored. The application will then send the log file to the web service to be stored and processed. An AI application will be created to monitor the log files and identify suspicious web activity using the Cloudflare API. When suspicious activity is detected on the server, the potentially dangerous IP will be blocked for twenty-four hours.

Applicable Courses from Iowa State University Curriculum

- COM S 227
- COM S 228
- COM S 252
- COM S 309
- COM S 311
- COM S 321
- COM S 329
- COM S 339
- COM S 352
- COM S 363
- CPR E 430
- CPR E 431
- COM S 474

New Skills/Knowledge acquired that was not taught in courses

- .NET servers
- Implementing code involving artificial intelligence
- ISS log files and general web traffic moderation

Table of Contents

List of figures/tables/symbols/definitions

- Definitions:
  - AI: Artificial Intelligence, a computer application that can learn, adapt to changing environments based on previous experiences.
  - API: Application Programming Interface, a software intermediary that allows two applications to talk to each other.
  - WAF: Web Application Firewall, an application firewall that applies a set of rules to an HTTP conversation.

1 Introduction

1.1 ACKNOWLEDGMENT

Technical advice and guidance were given by Dr. Yong Guan, Dr. Berk Glumezoglu, and Dr. Akhilesh Tyagi. Andrew Dakin provided technical advice and product specifications and a test domain from Cylosoft.

1.2 PROBLEM AND PROJECT STATEMENT

Problem Statement: Cylosoft is a company that designs, codes, and hosts websites. These websites are often probed and tested by bots, hackers, and spammers. The web servers at Cylosoft generated text log files as URLs are hit, and Cloudflare acts as a web application firewall (WAF). Cloudflare has both Cloudflare generated rules and customer-generated rules, and there are gaps in the rules that can be improved.

Solution Approach: This problem can be solved by creating a web server-side service that gathers logs and sends them in real-time to a cloud database. The database would then have code run against it, looking for potential bots, hackers, and spammers and generate firewall rules accordingly. The project's output will be real-time WAF rules generated by artificial intelligence based on web traffic patterns.

1.3 OPERATIONAL ENVIRONMENT

Our project's component is software that will run on hardware (computers, servers) located at the Cylosoft office. While we do not need to consider conditions such as heat, weather, and moisture for our product, we must ensure that every aspect of our code can run smoothly on Cylosoft's system.

1.4 REQUIREMENTS

Functional:

- Microsoft ISS will generate text-based log files of each web request.
- A console-based application will be created, which will find the current log file for the site that is being monitored. The application will then send the log file to the web service to be stored and processed.
- The Azure web service will process and store relevant information from IIS log files.
- The AI should have access to the processed IIS log data.
- An AI application will be created to monitor the log files and identify suspicious web activity using the Cloudflare API.
- When suspicious activity is detected on the server, the potentially dangerous IP will be blocked for twenty-four hours.

Non-Functional:

- The system should be able to perform network inspection for as much uptime as possible.
- Building and scaling the system should be easy to manage.
- The system should be able to accommodate large volumes of traffic.

- Access to the system should be restricted to only approved users.
- The system should block suspicious network traffic upon identifying it.

Environmental:

- Servers and databases should be able to function indoors.
- Rules and IP blocks of network firewalls should be updated per Cylosoft standards.
- Servers should function with web requests from any distance.

Economic:

- All testing should be completed using one domain/server, which will be provided by the client.
- All other design and implementation should be completed using development platforms and IDEs at no extra cost.
- The project should be completed with six team members, with each team member working three hours per week.
- The project should be designed and implemented by May of 2021.

1.5 Intended Users and Uses

Our product's intended end-user is Cylosoft, with the product operating on Cylosoft's existing software system. Our work is intended to be scalable to meet the needs of Cylosoft as they expand their software in the future.

1.6 Assumptions and Limitations

Assumptions:

- Supplies to accomplish Cylosoft will provide this system.
- Team members working on the project will be able to access the resources needed to accomplish their tasks.
- All equipment utilized will be in working condition.
- The scope of the project will not change for the duration of its life cycle.
- The project source code will be utilized after the project is completed.

Limitations:

- Existing network bandwidth can't be tied up by traffic generated from our project.
- Project planning shall be done by the end of the fall semester.
- The design and implementation of the project shall be done by the end of the spring semester.

1.7 Expected End Product and Deliverables

Expected End Product: ***Security Application with learning capabilities***

We will have a complete design document at the end of this project to help stakeholders understand the design process we used. A console-based application will read log files from Microsoft IIS. These log files will be parsed and fit the Azure web service database template. Each log file will be assigned to a corresponding website to ensure ease of accessibility of necessary information. The AI will be developed to have access to the processed IIS log information and help monitor them to identify suspicious activity. When the AI detects suspicious activity, it will remove the IP and block them from accessing the site for twenty-four hours.

Deliverables:

Status Reports-

- We will be providing bi-weekly reports on our progress and report any shifts in our timeline. The team will also give the date of all meetings and discussions.

User Manual-

- We will provide a README file as well as a LICENSE file contained within the GITHUB of the project. The README file will have detailed instructions on how to use the console application and make modifications to it. The LICENSE file will contain any legal documentation associated with Frameworks/IDE's/API's and other relevant material used in developing this project.

Project Design Document-

- We will provide a detailed plan and description of the process we used and the development of this application. This document will outline many things, including the Gantt chart, price cost, and technologies used and relevant to the project.

Azure Web Service-

- We will create an Azure web service that receives Microsoft IIS logs and stores the relevant data from them in a database.

Console Application-

- We will develop a console application to parse through Microsoft IIS log files and return them appropriately to fit Azure database fields. Log files will also be assigned their corresponding website so that they can be easily tracked

AI-

- Develop an AI that will find "obvious" threats that Cloudflare would typically not catch. These "obvious" threats will be recorded as rules to improve future responses to that type of attack. AI will then remove the threat.

Test-Case-

- We will provide an example of the log files association to a website and the proper formatting of Microsoft IIS log files inside the Azure database. Also, a simulated attack to show the performance of the AI in detecting and removing a threat.

2. Project Plan

2.1 Task Decomposition

Our initial approach to creating the solution was to break apart the problem into three smaller problems: a console application, a cloud web service, and an artificial intelligence algorithm to make Cloudflare rules. The console application will pull Microsoft IIS logs for a site being monitored and then send them to the web service. The web service will transform the records and fit them into our data structure. Our AI algorithm will then use the web service logs to generate Cloudflare WAF rules to be applied to the monitored site.

2.2 Risks And Risk Management/Mitigation

Main Tasks:

Research
Environment Setup
Console Application
Web Service
AI Application

Table 2.2 Main Tasks

Building console application - Risk Score: 0.6

- This task was given a risk score of 0.6 because there will be many different things to come together for this task to work correctly. This application will have to make connections between the Microsoft IIS log files and our web service. It also has to look at the IIS log files and extract meaningful information and translate that to a condensed log file to send over.

Building web service - Risk Score: 0.5

Building database service - Risk Score: 0.2

Building AI application - Risk Score: 0.9

- If the AI cannot identify malicious traffic to block correctly, we will need to find a different AI approach. This will likely involve figuring out another machine learning/AI learning algorithm that will identify it instead. This task has a high-risk score because our team has little experience with machine learning/AI and will likely have to change models.

Testing/Quality Assurance - Risk Score: 0.8

- Testing and making sure our systems work together correctly without failure might prove to be complicated. As testing

2.3 Project Proposed Milestones, Metrics, and Evaluation Criteria

Proposed Milestones:

Research-

Main Task:

- Research all things related to the project (Programming language, Frameworks, APIs)
    - Subtask:
        - Share information with the group and discuss the practicality

Environment Setup-

Main Task:

- Install IDE to be used for console application and AI and corresponding frameworks

Console Application -

Main Task:

- The console-based application will find the current log file for the site being monitored and send the log file to the web service
    - Subtasks:
        - Establish a connection between code and IIS log files
        - Establish a connection between code and web service
        - Find log file specific to the website
        - Parse log file for necessary information
        - Send parsed log file to web service

Web Service

Main Task:

- Azure web service will store relevant information from logs into the database
    - Subtask:
        - Establish a connection between code and web service
        - The parsed log file will store data in the correct fields in the database

AI-

Main Task:

- AI will detect attacks and write new rules that do not exist on the Cloudflare API to improve detection accuracy.
    - Subtasks:
        - AI would look through Cloudflare rules and remove the user if conditions met
        - Machine learning algorithm will add new rules not existing within Cloudflare

2.4 Project Timeline/Schedule

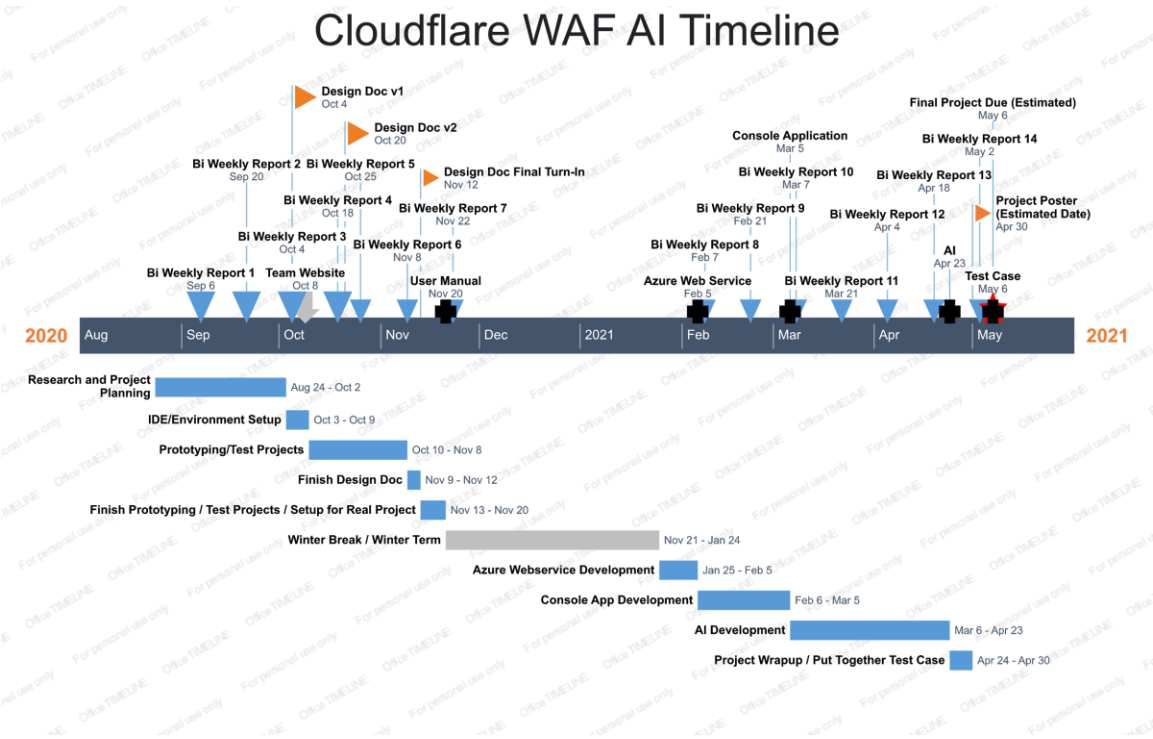

Figure 2.4 Project Schedule

Note on the Gantt chart: Listing all subtasks made the above chart unreadable, so each subtask's timelines are spelled out below.

Note on our development strategy: We plan on using an Agile development strategy with one-week sprints instead of the traditional two-week sprints. Our project has many small tasks that would be difficult to lump together in two-week sprints, and with this semester being remote, we were planning on meeting weekly from the beginning.

Breakdown Into Subtasks

- Research and Planning
  - Information gathered through research was shared with the group each week.
- IDE/Environment Setup
  - The IDE and necessary libraries are to be installed and set up for each developer by October 9th
- Prototyping / Test Projects
  - There should be a test project for the web service after the second week, a test console app after the third, and a test AI project by the final week of this task.
- Finish Design Doc
  - The design document should be completed and turned in by November 12th
- User Manual
  - There should be a rough user manual (subject to change throughout the project) by November 20th
- Azure web service
  - The first week will be dedicated to setting up a database
  - The second week will set up the database's automatic filling with test data sent to the web service.
- Console Application
  - The console application should be able to find log files specified by the end of the first week.
  - It should be able to parse the log files by week 3.
  - By the end of the fourth week, the console app should send the relevant data to the web service.
- AI
  - By the end of the second week, the AI application should set Cloudflare rules by manual input.
  - The rest of the time for AI will be used to develop the machine learning algorithm
- Project Wrap Up/ Test Case
  - By the project deadline, there should be a presentable test case for the program, and our project sponsors should fully understand how to duplicate our setup and take over the project from us.

2.5 Project Tracking Procedures

With the semester progressing, we have utilized different tools to help manage as our project grows. We are using slack to communicate with our clients and address any upcoming issues with our project. For project discussion among our group members, we heavily use discord to manage meetings and share resources that pertain to our project. It allows us to brainstorm and finalize ideas for our project. We have also set up a Github repository and GitLab manager to ready once we start development in our project. These repositories will help manage changes in our project among all of our members. We will all be able to make changes once our project starts without interfering with anyone's work. Our team also discussed the possibility of using Trello to help keep track of our project's achievements and significant milestones. From experience, we can all say that Trello is an excellent tool for development management. Aside from these useful management

tools, we can all express our team has utilized these to the fullest. We hope to come across any more valuable tools as we start development with our project!

2.6 Personnel Effort Requirements

TASK	ORDER	PERSON-HOURS
Research	1st	108
Environment Setup	2nd	18
Console Application	3rd	72
Web Service	4th	36
AI Application	5th	126

Table 2.1 Personnel Effort Requirements

All team members will initially take on each of the tasks. As the tasks get complex, team members will choose a specific job to focus on. The jobs are ordered because some functions depend on the previous implementation. For example, how the database is formatted depends on how the console application formats the IIS logs. The person-hours are estimates and will depend on how many obstacles our team runs into as we learn how to implement the project's tasks.

2.7 Other Resource Requirements

- Virtual environment for testing, provided by Cylosoft
- Microsoft Visual Studio, provided by Iowa State University