



Development Standards & Practices Used

Our project is strictly software, so no hardware will be used. We will be following a loose version of agile development and object-oriented programming. Part of our testing process will use end-user acceptance testing. Architecturally, our code will be using REST APIs and implementing good practices around database security.

Summary of Requirements

* Identify malicious IP addresses by analyzing IIS log files
* Use AI to block malicious IP addresses and store that data
* Create a .NET Core Console application that runs in Windows

Applicable Courses from Iowa State University Curriculum

* SE 329 - Project management
* COM S 363 - SQL database knowledge
* COM S 309 - Software Development Practices
* S E 339 - Software Architecture and Design

New Skills/Knowledge acquired that was not taught in courses

* .NET Core
* IP address blocking
* IIS logs
* Identifying malicious attacks (Cyber Security)

Table of Contents

[1 Introduction 4](#_heading=h.3znysh7)

[1.1](#_heading=h.2et92p0) Acknowledgement 4

[1.2](#_heading=h.tyjcwt) Problem and Project Statement 4

[1.3](#_heading=h.3dy6vkm) Operational Environment 4

[1.4](#_heading=h.1t3h5sf) Requirements 4

[1.5](#_heading=h.4d34og8) Intended Users and Uses 4

[1.6](#_heading=h.2s8eyo1) Assumptions and Limitations 5

[1.7](#_heading=h.17dp8vu) Expected End Product and Deliverables 5

[2](#_heading=h.3rdcrjn) Project Plan 5

[2.1 Task Decomposition 5](#_heading=h.26in1rg)

[2.2 Risks And Risk Management/Mitigation 6](#_heading=h.lnxbz9)

[2.3 Project Proposed Milestones, Metrics, and Evaluation Criteria 6](#_heading=h.35nkun2)

[2.4 Project Timeline/Schedule 6](#_heading=h.1ksv4uv)

[2.5 Project Tracking Procedures 6](#_heading=h.44sinio)

[2.6 Personnel Effort Requirements 7](#_heading=h.2jxsxqh)

[2.7 Other Resource Requirements 7](#_heading=h.z337ya)

[2.8 Financial Requirements 7](#_heading=h.3j2qqm3)

[3 Design 7](#_heading=h.1y810tw)

[3.1 Previous Work And Literature 7](#_heading=h.4i7ojhp)

[3.2](#_heading=h.2xcytpi) Design Thinking 7

[3.3](#_heading=h.1ci93xb) Proposed Design 7

[3.4 Technology Considerations 8](#_heading=h.3whwml4)

[3.5 Design Analysis 8](#_heading=h.2bn6wsx)

[3.6](#_heading=h.qsh70q) Development Process 8

[3.7](#_heading=h.3as4poj) Design Plan 8

[4 Testing 9](#_heading=h.147n2zr)

[4.1](#_heading=h.3o7alnk) Unit Testing 9

[4.2](#_heading=h.23ckvvd) Interface Testing 9

[4.3](#_heading=h.ihv636) Acceptance Testing 9

[4.4](#_heading=h.32hioqz) Results 9

[5 Implementation 10](#_heading=h.1hmsyys)

[6 Closing Material 10](#_heading=h.41mghml)

[6.1 Conclusion 10](#_heading=h.2grqrue)

[6.2 References 10](#_heading=h.vx1227)

[6.3 Appendices 10](#_heading=h.3fwokq0)

List of figures/tables/symbols/definitions

Figure 2.4.1 Semester 1 Gantt Chart

Figure 2.4.2 Semester 2 Gantt Chart

# 1 Introduction

## Acknowledgement

Cylosoft, and particularly Andrew Dakin, has provided us with significant resources necessary to complete this project including, sample logs, and technical experience. We have also received help from our ISU advisor, Doug Jones.

## Problem and Project Statement

Cylosoft has a CMS Platform used to host many customer websites. On a regular basis, the websites are probed and tested for issues by bots and hackers. SmartBlock is written to allow the website to actively detect and stop these attacks. This involves writing an application to read web server traffic logs in real-time. The logs are scanned by AI and used to block bad traffic. The AI can peer across all the CMS sites and then write IP blocking rules to eliminate the attackers attempts.

## Operational Environment

Our product is expected to be able to handle multiple sites on a single server functioning on a Windows Machine. The data collected from our application will be held in an accessible database for Cylosofot to monitor. There are no physical components to our project, everything is digital.

## Requirements

Functional Requirements

* Application should recognize bad web traffic based on definitions provided by Cylosoft
* Application should block bad web traffic through IIS config changes
* Application should be able to read Microsoft IIS log files
* Application should keep a record of IPs that were blocked
* Application should process multiple sites on a single server
* Application should be have a perceptive AI

Economics Requirements

* The design will take no longer than 500 person-hours
* Application development should take approximately 1000 person-hours

Environmental Requirements

* Application should run on a Windows Machine
* Application should use an MSSql Database
* Application should use .NET Core

## Intended Users and Uses

Our application’s main end user is to be our client, Cylosoft. Cylosoft intends to use our application to help them cut down on resources that are being held up by manually identifying and blocking bad web traffic that accesses their product. In a discussion with our client, it was decided that, based on the final deliverable, we may make our code open-source and available to anyone.

## Assumptions and Limitations

Assumptions

* Server provided to run the application on.
* Application will run in the background.
* AI will be able to be used by anyone, not just the company.
* Will be provided with access to real-time IIS logs.
* If used by other companies, their files are formatted the same as the company we are developing the application for.

Limitations

* Not checking for a spoofed location.
* Not checking for changed IP addresses.
* Will only block IP addresses based on suspicious activity shown in IIS logs, and cannot guarantee to prevent all attacks.
* Not testing integrity of data.

## Expected End Product and Deliverables

The primary goal of our project is to aid Cylosoft in preventing cyberattacks. This will manifest itself in a piece of software that will look through Cylosoft’s server logs, detect suspicious activity, and take appropriate action to prevent the source of the suspicious activity from accessing Cylosoft’s website. Thus, the main product that we will be producing for Cylosoft is a web-attack blocking AI software.

# Project Plan

## 2.1 Task Decomposition

Our project can be split up into two stages where each stage has their own tasks and subtasks.

Stage One: Create a .Net Console application that reads IIS log files

1. Create a MySQL database to host the storage of log data
   1. Gather sample log files to create fields for the necessary pieces of a log file.
   2. Implement restricted user permissions to access database data.
2. Create a private Github repo for future open source compatibility.
   1. Add any sensitive material and production code as we go here.
3. Create a .Net Core application
4. Connect the MySQL database with the .Net Core application
   1. Enable the application to store new entries into the database.
5. Connect the .Net Core application with IIS logging to read in data
   1. Create a scalable algorithm to handle various fluxes of logs being received.
6. Determine what qualifies as a bad or good log file and the related signatures of each.

Stage Two: Design an AI that will block bad incoming traffic

1. Implement an AI that dynamically analyzes the validity of incoming traffic.
2. Block a specific IP address and record any identifying signatures to feed back into the AI.

## 2.2 Risks And Risk Management/Mitigation

1. Create a MySQL database to host the storage of log data: 0.1.
   1. The risk for this step is 0.1 because our team is familiar with MySQL database and it is a good reliable tool that we can easily use.
2. Github: 0.1
   1. The risk for this step is 0.1 because we have all used git before and it is the version control system our team is the most familiar with.
3. .NET Core Application: 0.3
   1. The risk for this step is 0.3 as not everyone has experience with .NET console applications and there is a chance it doesn’t work.
4. Connect with IIS logs: 0.3
   1. The risk for this step is 0.3 because while we have not worked with IIS logs before, they should be fairly easy to connect to the .NET core application.
5. Implement AI: 0.4
   1. The risk for this step is 0.4 as the AI should be difficult to make and implement, but it is a necessary part of our project.
6. Block IP Addresses: 0.3
   1. The risk for this step is 0.3 because blocking an IP address should be fairly easy but there is a risk whoever was blocked just changes their IP address.

## 2.3 Project Proposed Milestones, Metrics, and Evaluation Criteria

Criteria:

The two main criteria for our web attack blocking software will be accuracy and efficiency.

In terms of accuracy, it is important that our product is not missing any potential malicious actors that are attempting to access Cylosoft’s website. Therefore, one of the main metrics we will measure is attack classification correctness. This can be defined as the percentage of attacks we are classifying as malicious activity compared to the actual percentage of attacks that are malicious. This criteria is the main focus for the project.

The second criterion is how fast and efficient our software is detecting these attacks. Speed and efficiency are key pieces because if a malicious actor is attempting to access a site, one would not want the actor performing potentially dangerous activity while waiting to be blocked. Thus, we will measure how long it takes for our AI to detect and block the actor in some standard time measurement such as milliseconds.

Metrics:

1. Attack classification correctness (% of correct classifications)
2. Speed / detection time (m/s)

Milestones:

1. Setup fundamental resources.
2. Ability to parse IIS logs and extract relevant information.
3. Detect an attack from IIS log successfully.
4. Implement AI using the algorithm to detect attack.
   1. Detect attack with >80% accuracy.
5. Block an IP address automatically after access is categorized based from extracted logs.
   1. Classification with >95% accuracy.
   2. Nearly instantaneous detection time.

## 2.4 Project Timeline/Schedule

For our scheduling plans, please refer to the documentation that is linked below. This link will take you to our [Gantt chart](https://docs.google.com/spreadsheets/d/1XKg0tS8S4s9vkntKFb-iq3ANwFSfIdAKgnCr7fy26fw/edit?usp=sharing) that covers both the fall 2020 and spring 2021 semesters.

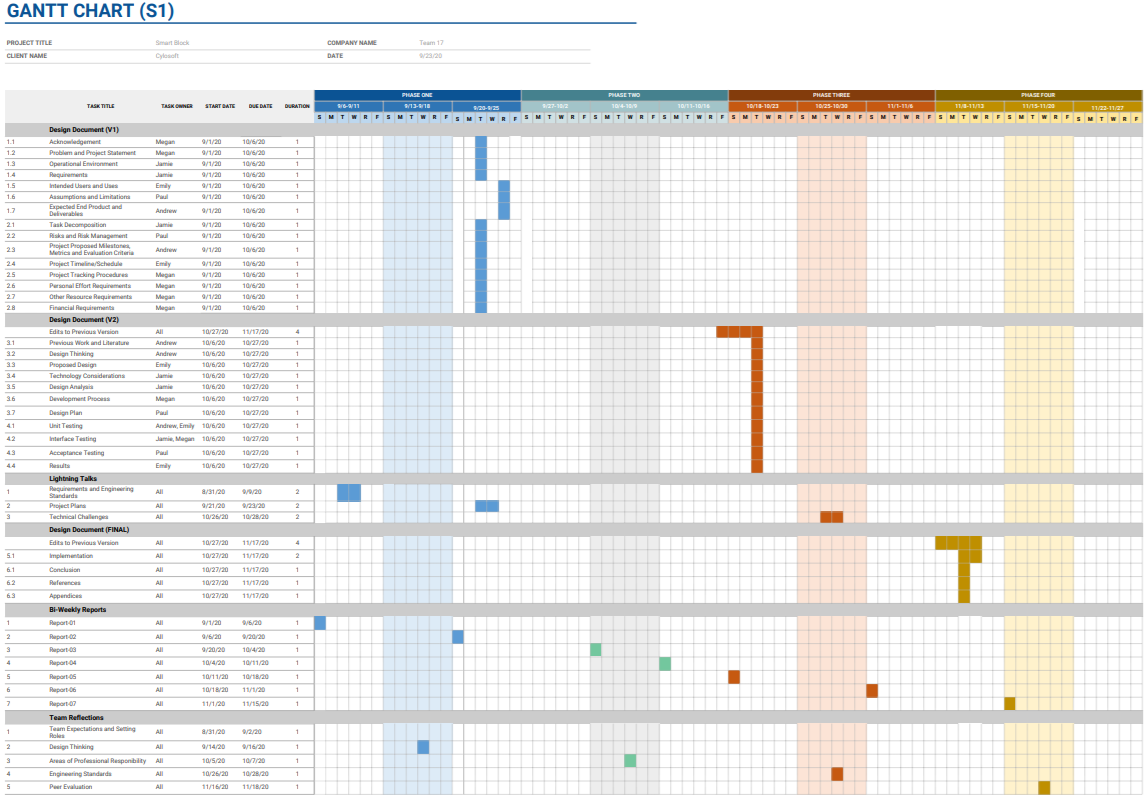


Figure 2.4.1 Semester 1 Gantt Chart

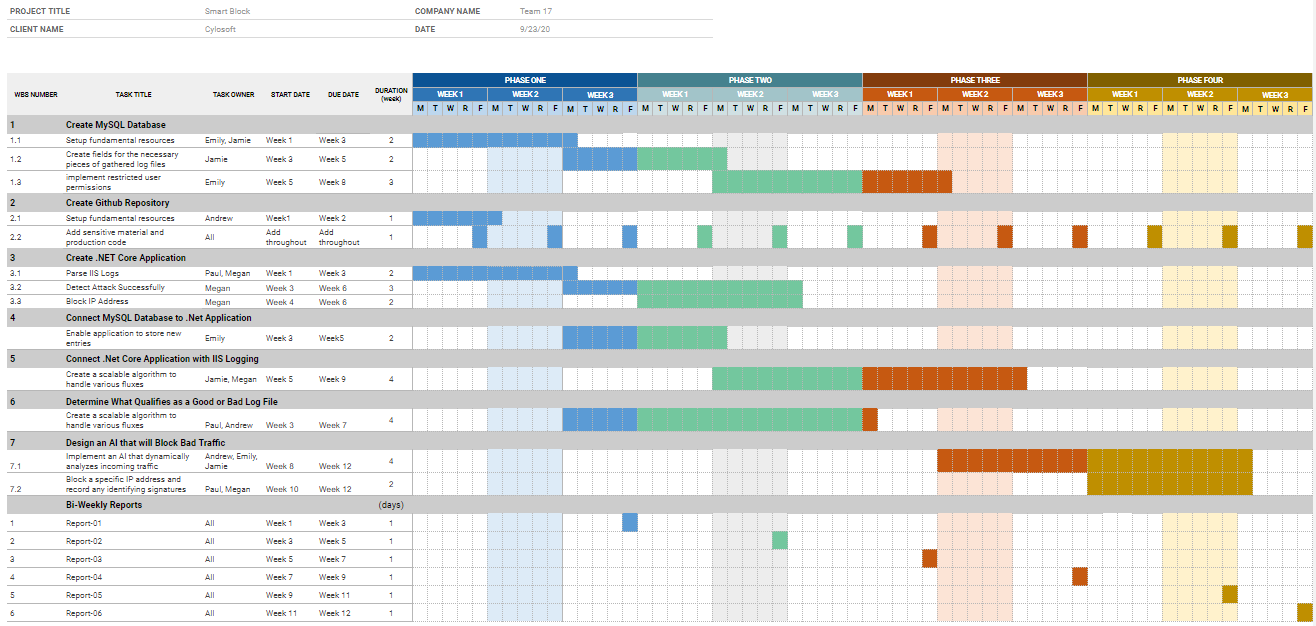


Figure 2.4.2 Semester 2 Gantt Chart

## 2.5 Project Tracking Procedures

Our team will be using GitHub to store all of our code for this project. We will use GitHub issues with labels for code tracking. This will allow us to keep track of everything that we need for the project in one place, while making it easier to link to existing code for clarifications. It will also allow us to keep track of why we are doing certain things (e.g. bugs, feature work, specific task). We will also be using Slack and WebEx to communicate with each other to define all of the necessary aspects of the project. This could include determining necessary features from the clients, or breaking up work between the team members.

## 2.6 Personnel Effort Requirements

|  |  |
| --- | --- |
| Task | Hours |
| Create a MySQL database to host the storage of log data | 15 |
| Gather sample log files to create fields for the necessary pieces of a log file. | 10 |
| Implement restricted user permissions to access database data. | 10 |
| Create a private Github repo for future open source compatibility. | 5 |
| Create a .Net Core application | 30 |
| Connect the MySQL database with the .Net Core application | 35 |
| Enable the application to store new entries into the database | 30 |
| Connect the .Net Core application with IIS logging to read in data | 40 |
| Create a scalable algorithm to handle various fluxes of logs being received. | 75 |
| Determine what qualifies as a bad or good log file and the related signatures of each. | 25 |
| Implement an AI that dynamically analyzes the validity of incoming traffic. | 100 |
| Block a specific IP address and record any identifying signatures to feed back into the AI. | 25 |
| Total | 400 |

## 2.7 Other Resource Requirements

In order to complete the project, our team will need a variety of sample logs to test our application. These logs will need to have a combination of normal traffic and malicious traffic, with the malicious traffic marked (including why they are believed to be malicious) so that we know which scenarios to block. We will also need some database space so that we can keep a record of all logs that were blocked. Once the project is ready to be tested, we will need access to a dev environment so that we can test blocking traffic, without potentially negatively impacting the prod environment.

## 2.8 Financial Requirements

This project does not require any hardware, and we will be using all on prem databases, so there will not be any financial obligations for this project.

# 

# 3 Design

## 3.1 Previous Work And Literature

Include relevant background/literature review for the project

–  If similar products exist in the market, describe what has already been done

–  If you are following previous work, cite that and discuss the **advantages/shortcomings**

–  Note that while you are not expected to “compete” with other existing products / research groups, you should be able to differentiate your project from what is available

Detail any similar products or research done on this topic previously. Please cite your sources and include them in your references. All figures must be captioned and referenced in your text.

## Design Thinking

Detail any design thinking driven design “define” aspects that shape your design. Enumerate some of the other design choices that came up in your design thinking “ideate” phase.

## Proposed Design

Include any/all possible methods of approach to solving the problem:

* Discuss what you have done so far – what have you tried/implemented/tested?
* Some discussion of how this design satisfies the **functional and non-functional requirements** of the project.
* If any **standards** are relevant to your project (e.g. IEEE standards, NIST standards) discuss the applicability of those standards here
* This design description should be in **sufficient detail** that another team of engineers can look through it and implement it.

## 3.4 Technology Considerations

Highlight the strengths, weakness, and trade‐offs made in technology available.

Discuss possible solutions and design alternatives

## 3.5 Design Analysis

–  Did your proposed design from 3.3 work? Why or why not?

–  What are your observations, thoughts, and ideas to modify or iterate over the design?

## Development Process

Discuss what development process you are following with a rationale for it – Waterfall, TDD, Agile. Note that this is not necessarily only for software projects. Development processes are applicable for all design projects.

## Design Plan

Describe a design plan with respect to use-cases within the context of requirements, modules in your design (dependency/concurrency of modules through a module diagram, interfaces, architectural overview), module constraints tied to requirements.

## 

## 

# 

# 

# 4 Testing

Testing is an **extremely** important component of most projects, whether it involves a circuit, a process, or software.

1. Define the needed types of tests (unit testing for modules, integrity testing for interfaces, user-study or acceptance testing for functional and non-functional requirements).  
2. Define/identify the individual items/units and interfaces to be tested.  
3. Define, design, and develop the actual test cases.  
4. Determine the anticipated test results for each test case

5. Perform the actual tests.  
6. Evaluate the actual test results.  
7. Make the necessary changes to the product being tested

8. Perform any necessary retesting  
9. Document the entire testing process and its results

Include Functional and Non-Functional Testing, Modeling and Simulations, challenges you have determined.

## Unit Testing

– Discuss any hardware/software units being tested in isolation

## Interface Testing

–  Discuss how the composition of two or more units (interfaces) are to be tested. Enumerate all the relevant interfaces in your design.

## Acceptance Testing

How will you demonstrate that the design requirements, both functional and non-functional are being met? How would you involve your client in the acceptance testing?

## Results

– List and explain any and all results obtained so far during the testing phase

* Include failures and successes
* Explain what you learned and how you are planning to change the design iteratively as you progress with your project
* If you are including figures, please include captions and cite it in the text

# 

# 5 Implementation

Describe any (preliminary) implementation plan for the next semester for your proposed design in 3.3.

# 

# 6 Closing Material

## 6.1 Conclusion

Summarize the work you have done so far. Briefly reiterate your goals. Then, reiterate the best plan of action (or solution) to achieving your goals and indicate why this surpasses all other possible solutions tested.

## 6.2 References

List technical references and related work / market survey references. Do professional citation style (ex. IEEE).

## 6.3 Appendices

Any additional information that would be helpful to the evaluation of your design document.

If you have any large graphs, tables, or similar data that does not directly pertain to the problem but helps support it, include it here. This would also be a good area to include hardware/software manuals used. May include CAD files, circuit schematics, layout etc,. PCB testing issues etc., Software bugs etc.