Cloudflare WAF Scanner

Team Information – sdmay21-16

Client
Cylosoft - Andrew Dakin

Faculty Advisors
Dr. Berk Gulmeszoglu
Dr. Yong Guan

Team Members

Ryan Burgett	rdb@iastate.edu	SE
Ben Feddersen	brfedd@iastate.edu	SE
Jordan Heim	jcheim@iastate.edu	CYBSC
Pressiian Iskrenov	presiian@iastate.edu	SE
Giovanni Mejia	gsmejia@iastate.edu	CPRE
Eric Reuss	ewreuss@iastate.edu	SE

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 generate text log files as URLs are hit, and Cloudflare acts as a web application firewall (WAF). Cylosoft uses Cloudflare as its firewall, which has both Cloudflare generated rules and customer-generated rules, and there are gaps in the rules that can be improved.

Proposed Solution

Parse through Microsoft IIS log files and insert log files into Azure Database to then be scanned every two minutes by multiple scanners which return high risk IP addresses to be used with CloudFlare API to mitigate risk of attack.

Design Requirements

Functional Requirements

-Web server and console application will generate log files and web requests.
-Database will store log files.
-Azure webservice will process log files.
-Database scanner will monitor the log files and update usage rules to deny access to suspicious users.
-Web Application will provide a user-friendly way to interface with log data.

Non-Functional Requirements

-Azure web service will be always active.
-Database and console application will not be limited by the quantity of logs processed.
-Security checkpoints will be in place to limit access to data.
-Cloudflare module will block suspicious network traffic.
-Web Application will require authentication.

Intended Uses/Users

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.

Engineering Standards

-**IEEE 1028-1997** Standard for Software Unit Testing
-**IEEE 12207-2017** Software Life Cycle Process
-**IEEE 16326-2009** Project Management

Engineering Constraints

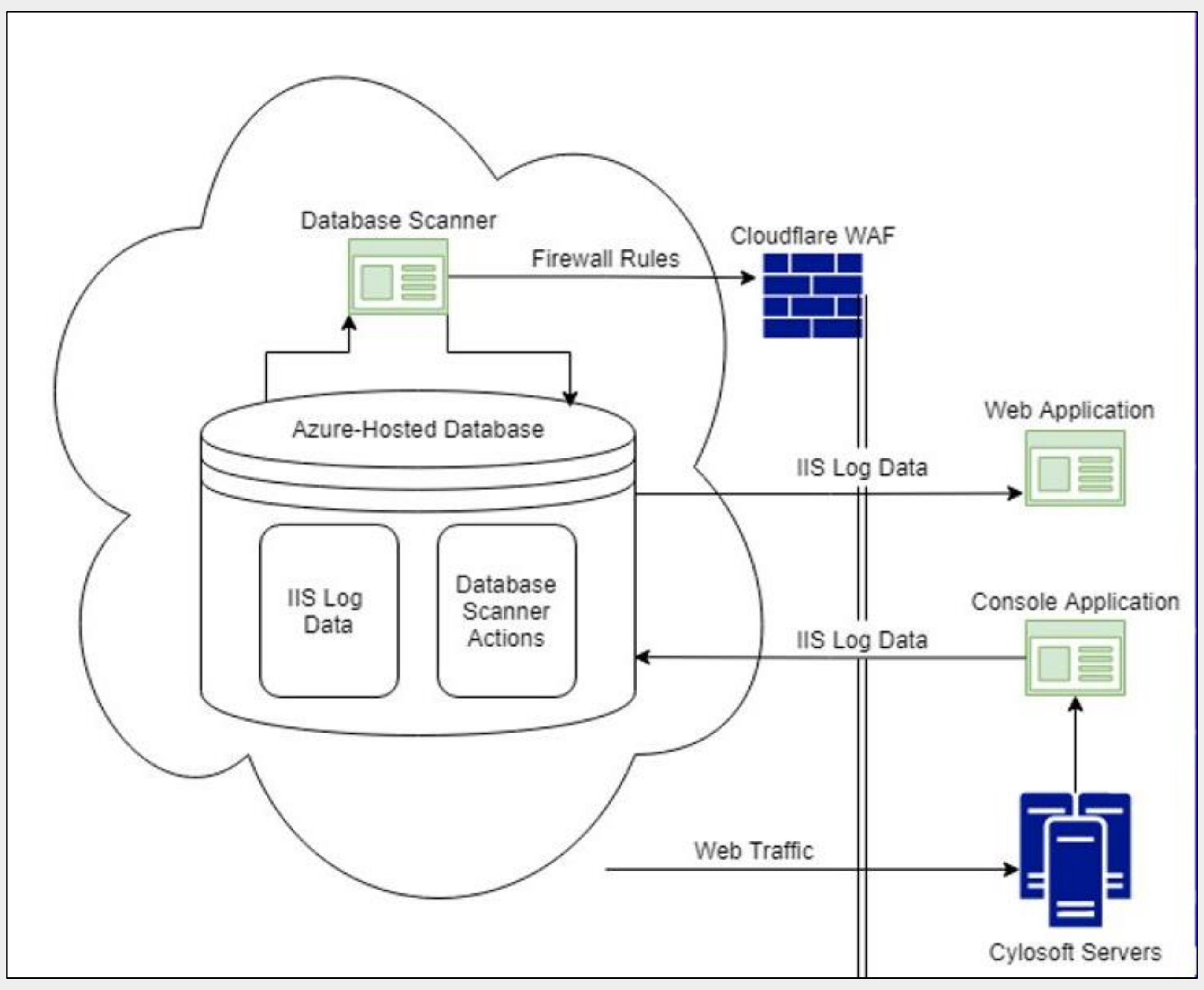
-Covid
-Team Size
-Legal Obligations
-Time Constraint
-Acceptance Testing

Software Modules/Technologies

-Visual Studio IDE
-C# (.NET Framework)
-Azure Database
-MYSQL
-Cloudflare WAF (API)

Design Approach

Software Architecture Sketch



Technical Details

Web Server

The web server contains the website our project oversees protecting. Our project should be able to work for a variety of different web servers.

Console Application

Responsible for finding, parsing, and sending relevant information from IIS log files to the Azure web service. This application will make use of REST protocols for sending data.

Azure web service

Consists of a database for storing IIS log information. This will make use of REST protocols for transferring data between the database, the console application, and the Scanner.

Database scanner

Responsible for analyzing the relevant data from the database and creating new Cloudflare rules based on that data.

Cloudflare WAF

The firewall which will block unwanted traffic from reaching the web server. Rules will be updated in real time by our application through its API.

Web application

The web application will be responsible for providing a user-friendly interface for our client to easily view and filter the contents of our database.

Security Concerns

There are limitations with our scanners, there are many different types of attacks, and we have only account for some, these scanners will need to be updated and maintained.

Testing

Console Application
- -Used sample log files provided by our client.
- -Used local IIS to replicate production environment.
- -Initially tested on low risk/low volume server.

Web Application
- -Used chrome debugging console to identify dependency errors.
- -Connected Web Application to live database.
- -Ensured testing environment was identical to the production environment.

Database Scanner
- -Used Log files with known attacks provided by our client.
- -Pulled Data from existing database.
- -Connected database scanner to live database to load data from console application.

Acknowledgements

We would like to thank the Cylosoft team for giving us the opportunity to work on such an interesting project. We would also like to thank our faculty advisors Berk and Yong, and our client Andrew for all of their amazing support and insights.