

**Project Archive:**

<https://github.com/endpointe/watchlog/tree/main>

<https://watchlog.wiki>

**Alvin Johns**

**June 7, 2024**

# Contents

<b>1</b>	<b>Foreward: Release Notes</b>	<b>2</b>
1.1	Encrypt the data before sending to compression . . . . .	2
1.2	Compress the data before sending to the storage-controller . . . . .	2
1.3	Enable TLS + Certificates and Key Rotation Playbook . . . . .	2
1.4	Utilize the available signal handlers . . . . .	2
1.5	Create a TUI/GUI for the user . . . . .	2
1.6	Windows support . . . . .	2
1.7	Testing . . . . .	2
<b>2</b>	<b>Personnel and Legacy</b>	<b>3</b>
2.1	Who requested the project? . . . . .	3
2.2	Why was it requested? . . . . .	3
2.3	Who were the project partners? . . . . .	3
2.4	Who are the members of your team? . . . . .	3
2.5	What were the roles of the project partner(s)? . . . . .	3
<b>3</b>	<b>PRD/SDA/SDP Documents</b>	<b>4</b>
<b>4</b>	<b>User Guide Document/Handoff Documents</b>	<b>4</b>
<b>5</b>	<b>Technical Resources</b>	<b>4</b>
<b>6</b>	<b>Future Direction</b>	<b>5</b>
<b>7</b>	<b>Additional Information</b>	<b>5</b>

# 1 Foreward: Release Notes

The following is a list of todo's:

- + Encrypt the data before sending to compression
- + Compress the data before sending to the storage-controller
- + Enable TLS + Certificates and Key Rotation Playbook
- + Utilize the available signal handlers
- + Create a TUI/GUI for the user
- + Windows support
- + Testing

## 1.1 Encrypt the data before sending to compression

In `src/main.rs:encrypt(...)`, the `openssl` library handles encryption. It may be helpful to roll your own encryption as either a learning experience or to avoid the overhead of the `openssl` library. If you choose to use the `openssl` library, make sure to statically link the library to the binary. This has been set by using the 'vendored' option in the `Cargo.toml` file.

## 1.2 Compress the data before sending to the storage-controller

As with encryption, you can roll your own or use an existing library.

## 1.3 Enable TLS + Certificates and Key Rotation Playbook

Provide TLS support between `watchlog` client and storage-controller. Incorporate certificates and a key rotation playbook.

## 1.4 Utilize the available signal handlers

The signal handlers found in `src/main.rs:unix_app()` provide the ability for the application to be controlled from within the threads. In `src/main.rs:watch_logs()`, the `terminate_flag` can be passed into the `src/main.rs:collector(...)` function to provide better control of logs and the overall application.

## 1.5 Create a TUI/GUI for the user

The current application is a command-line application. A TUI/GUI would provide a better user experience. The TUI/GUI could be created using the 'ratatui' library. If creating a GUI, Tauri is a good option.

## 1.6 Windows support

Knowledge of the Windows API is required to port the application to Windows. The application is currently only supported on Unix-based systems. This is a great opportunity to extend the user/customer base.

## 1.7 Testing

If and when `tls`, encryption, and compression are implemented, testing will be required to ensure the application is functioning as expected.

## **2 Personnel and Legacy**

### **2.1 Who requested the project?**

This project was requested as part of the Oregon State University Capstone program. The staff requested the project to provide an alternative solution for capturing system logs.

### **2.2 Why was it requested?**

This project demonstrates the importance of maintaining observability in a system while reducing the overhead of the tools used to provide that observability.

### **2.3 Who were the project partners?**

The project partners were: Ivan Chan, Joseph Murche, Kevin Huynh, and Alvin Johns.

After a series of unfortunate events, I, Alvin Johns(endpointe), felt the need to move forward with the project in a separate repository. The project partners were notified of the move and were given the opportunity to continue with the project, or work on another project.

Link: <https://github.com/SecurityLogMiner/log-collection-client>

### **2.4 Who are the members of your team?**

The members of the WatchLog team are: Alvin Johns and, hopefully, future team members.

### **2.5 What were the roles of the project partner(s)?**

Alvin Johns: Project Manager, Developer, and Documenter

### 3 PRD/SDA/SDP Documents

PRD README: <https://github.com/endpointe/watchlog/blob/main/docs/PRD.md>

PRD PDF: <https://github.com/endpointe/watchlog/blob/main/docs/PRD.pdf>

SDA README: <https://github.com/endpointe/watchlog/blob/main/docs/SDA.md>

SDA PDF: <https://github.com/endpointe/watchlog/blob/main/docs/SDA.pdf>

SDP README: <https://github.com/endpointe/watchlog/blob/main/docs/SDP.md>

SDP PDF: <https://github.com/endpointe/watchlog/blob/main/docs/SDP.pdf>

### 4 User Guide Document/Handoff Documents

- User Guide README: <https://github.com/endpointe/watchlog/blob/main/README.md>

### 5 Technical Resources

Building a TUI or GUI:

Ratatui: <https://ratatui.rs/>

Tauri: <https://tauri.app/>

Building dashboards:

Wazuh: <https://wazuh.com/>

Wazuh Query Language (WQL):

<https://documentation.wazuh.com/current/user-manual/wazuh-dashboard/queries.html>

Elastic: <https://www.elastic.co/>

Kibana Query Language (KQL):

<https://www.elastic.co/guide/en/kibana/current/introduction.html>

Vector: <https://vector.dev/>

Building a Windows application with Rust:

<https://learn.microsoft.com/en-us/windows/dev-environment/rust/rust-for-windows>

## 6 Future Direction

Given that the base branch routes event data in plaintext (or encrypted) to the storage-controller, there is an opportunity to ciphon the data into an analytics module. This module could serve as the foundation for building a dashboard, making use of existing query languages (KQL/WQL) or perhaps building a custom one. Detection of abnormal behavior could generate alerts, notifying the appropriate users from which that event originated from.

## 7 Additional Information

Release Candidate 2 was created before the issue with encryption and compression was discovered. The directory structure on the storate-controller side was also created after the video was recorded. In the video, I mentioned that this step would take me about a day to get working and tested, which it did. The test named 'test\_send\_data()' function makes this process simple and the requirements are described within that test function as well as below:.

test\_send\_data()' function requirements:

- The storage-controller must be running
- dummy.data must have data
- logs/name/yyyy-mm-dd/hh-hh must exist on the storage-controller directory

RC2: [https://media.oregonstate.edu/media/t/1\\_s3gs95f1](https://media.oregonstate.edu/media/t/1_s3gs95f1)  
consenting