# SPECYFIKACJA WYMAGAŃ

## WYMAGANIA UŻYTKOWNIKA

1. Produkt musi:
   1. realizować funkcjonalność przełącznika sieciowego lub routera IP
   2. obsługiwać dwa protokoły- MELSEC SLMP oraz Modbus TCP
   3. posiadać interfejs WWW umożliwiający: zarządzanie regułami, analizę zdarzeń, definicję reguł na poziomie rejestrów.
   4. posiadać domenę strony internetowej w protokole https
2. Produkt powinien:
   1. pracować w środowisku powszechnie używanego systemu operacyjnego
   2. zapisywać historię przepływu pakietów
3. Produkt mógłby:
   1. mógłby formatować zapisywane dane do postaci raportu

## WYMAGANIA BIZNESOWE

1. Produkt musi:
   1. Stanowić dedykowane rozwiązanie problemu filtrowania ruchu protokołów przemysłowych
   2. Powstać na bazie komputera Raspberry Pi
   3. być rozbudowany o dodatkowe moduły hardware by zapewnić krytyczne funkcjonalności
2. Produkt powinien:
   1. operować na standardowych, bibliotekach Open Source
3. Produkt mógłby:
   1. być dystrybuowany Open Source

## WYMAGANIA SYSTEMOWE

1. System musi:
   1. Zostać pomyślnie uruchomiony na płytce Raspberry Pi w systemie raspbian
   2. Zawierać interfejs w postaci strony https z domeną, umożliwiający wprowadzanie i wyboru reguł
   3. Posiadać implementację obsługi protokołów MELSEC SLMP i Modbus TCP
   4. Posiadać moduł filtrujący ruch protokołów przemysłowych
   5. Zawierać lokalnie uruchomiony serwer webowy
2. System powinien:
   1. Umożliwiać zapisywanie historii zapisu pakietów
3. System mógłby:
   1. Mógłby formatować dane historii zapisu pakietów do postaci raportów

# Zastosowane szablony architektoniczne

## Moduł CONF

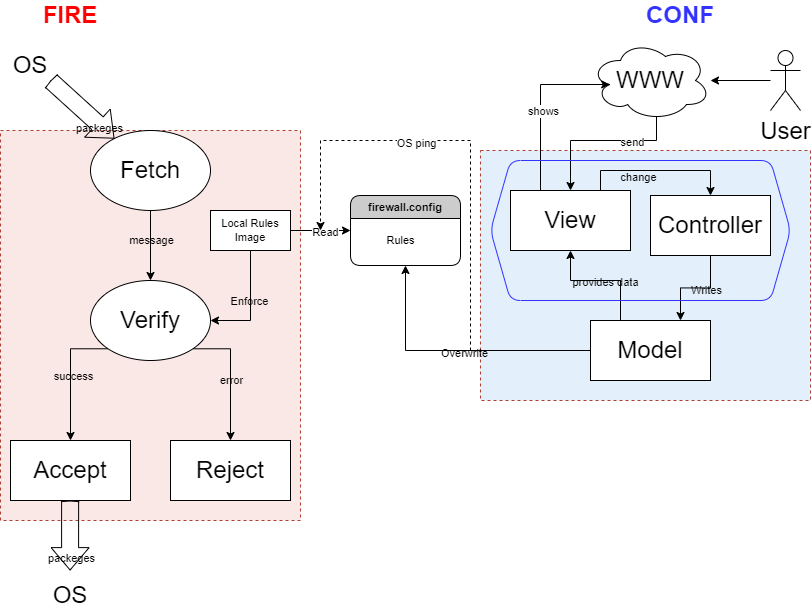
Architektura jest zrealizowana na bazie MVC zgodnie z poniższą tabelą:

|  |  |
| --- | --- |
|  | CONF |
| Model | Obsługuje serwer internetowy, przechwytuje komendy użytkownika. |
| View | Interpretuje komendę na konfigurację, przeszukuje na obecność błędów. |
| Controller | Pisze/Czyta zadaną konfigurację do pliku konfiguracyjnego |

## Moduł fire

Jest zrealizowany na bazie architektury filtrów i potoków zgodnie z poniższą tabelą:

|  |  |  |
| --- | --- | --- |
|  |  | FIRE |
| INPUT | | Kolejka systemowa nadchodzących pakietów |
| F1 | **Fetch** | Ściąga pakiety z kolejki i składa w pełne wiadomości |
| F2 | **Verify** | Weryfikuje poprawność wiadomości z regułami |
| OUTPUT | **Accept** | Przesyła pakiety składające się na wiadomość dalej |
| **Reject** | Wysyła z powrotem sygnał ICMP z kodem błędu |



# Kluczowe elementy struktury i ich interfejsy

## Moduł FIRE – Python3

Moduł FIRE odpowiada za funkcjonalną część implementowanego firewall’a. Przechwytuje on pakiety z kolejki systemowej za pomocą pakietu Netfilter. Następnie analizuje on pakiety pod kątem reguł zadanych w pliku konfiguracyjnym i podejmuje decyzję o bądź przesłaniu dalej, bądź o opuszczeniu pakietu.

Poniższa tabela przedstawia interfejs modułu FIRE:

|  |  |  |
| --- | --- | --- |
| Metoda | Zwraca | Działanie |
| ReadMessage() | Powodzenie – Strukturę reprezentującą pełną wiadomość, wraz z oryginalnymi pakietami na nią się składającymi Porażka – kod błędu | Czyta pakiety z kolejki systemowej, póki nie poskłada z nich wiadomości. |
| AnalyseMessage(mes) | TRUE -> ACCEPT FALSE -> REJECT | Analizuje wiadomość pod kątem obecnego zbioru reguł. |
| AcceptMessage(mes) | Powodzenie – 0  Porażka – Kod Błędu | Przepuszcza pakiety składające się na wiadomość dalej. |
| RejectMessage(mes) | Powodzenie – 0  Porażka – Kod Błędu | Odrzuca pakiety, wysyłając nadawcy pakiet ICMP. |
| UpdateConfig(dir) | Powodzenie – 0  Porażka – Kod Błędu | Aktualizuje zbiór reguł na podstawie obecnej zawartości pliku konfiguracyjnego ze ścieżki |

## MODUŁ CONF – PYTHON3

Moduł CONF odpowiada za możliwość konfiguracji implementowanego firewall’a. Wystawia on interfejs sieciowy umożliwiający użytkownikowi definicję nowych bądź modyfikację/usunięcie starych reguł. Po przeparsowaniu legalności działań użytkownika, moduł modyfikuje plik konfiguracyjny.

Poniższa tabela przedstawia interfejsy modułu CONF:

|  |  |  |
| --- | --- | --- |
| Metoda | Zwraca | Działanie |
| OPENWEB() | Powodzenie – 0 Porażka – kod błędu | Wystawia interfejs WWW |
| CLOSEWEB() | Powodzenie – 0 Porażka – kod błędu | Zamyka interfejs WWW |
| ANALYSERULE(mes) | Powodzenie – struktura Rule  Porażka – Kod Błędu | Interpretuje komunikat ze strony interfejsu WWW w nową zasadę i analizuje jej legalność. |
| WRITERULE(RULE, DIR) | Powodzenie – 0  Porażka – Kod Błędu | Modyfikuje plik konfiguracyjny pisząc do niego legalną regułę zadaną przez użytkownika |
| READConfig(dir) | Powodzenie – [Rule]  Porażka – Kod Błędu | Czyta z zadanego pliku konfiguracyjnego zbiór reguł i parsuje je do tablicy struktur Rule |

## PLIK KONFIGURACYJNY – FIREWALL.CONF

Plik konfiguracyjny odpowiada za zestaw reguł stosowanych przez moduł FIRE. Działa on w trybie White Listy, a więc zdefiniowane przez owe reguły pakiety są przepuszczane a reszta blokowana. W osobnych wierszach trzymane są definicje reguł w postaci przedstawionej w poniższej tabeli:

|  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- |
| RuleID | Name | Protocol | Profile | Direction | Analysed param | Expected Val |
| SHORT | VARCHAR(50) | [MODBUS/SLPM] | SHORT | [IN/OUT/BOTH] | VARCHAR(10) | VARCHAR(50) |

Powyższa reprezentacja może się zmienić podczas implementacji w zależności od wymagań struktur programowych.

# Interakcje pomiędzy elementami

## CONF – FIRE

Celem zastosowanej architektury była jak największa separacja modułu FIRE od CONF, aby w razie niesprawności narażonej zewnętrznie usługi sieciowej zapewnianej przez CONF, nie wyłączyć modułu FIRE odpowiadającego za bezpieczeństwo.

Mając powyższe na względzie, jedyną spodziewaną interakcją, jest wysłanie sygnału PING przez moduł CONF przy zmianie pliku konfiguracyjnego.

## PLIK KONFIGURACYJNY – CONF

Plik konfiguracyjny jest czytany i pisany przez moduł CONF.

Czytanie pliku konfiguracyjnego jest na potrzeby realizacji wirtualnego środowiska zasad wewnątrz modułu, które następnie są prezentowane jak w stanie obecnym na interfejsie WWW.

Pisanie do pliku konfiguracyjnego odbywa się na żądanie autoryzowanego użytkownika, który poprzez zmianę w interfejsie WWW modyfikuje zestaw reguł. Po wykryciu takiego działania, plik jest modyfikowany by odpowiadał wymaganiom obecnym.

## FIRE – PLIK KONFIGURACYJNY

Plik konfiguracyjny jest czytany przez moduł FIRE.

Po otrzymaniu ze strony systemu operacyjnego informacji o zmianie zawartości pliku konfiguracyjnego, moduł FIRE wczytuje nowy zbiór reguł. Po ich przeparsowaniu, natychmiastowo się do nich stosuje.

# Wyjaśnienie istoty przyjętych rozwiązań

# Określenie podstawowych mechanizmów technicznych

## Sprzęt

## Systemy operacyjne

## Serwer aplikacyjny

## Inne

## System raportowania

## System analityczny

## Mechanizmy zarządzania

## Mechanizmy bezpieczeństwa

# ANALIZA PRZYPADKÓW UŻYCIA: PRZEGLĄDANIE REGUŁ

## PB1: PRZEGLĄDANIE REGUŁ STANDARDOWYCH

Aktorzy: administrator

Scenariusz główny:

1. System sprawdza tożsamość i uprawnienia użytkownika
2. System wyświetla listę zdefiniowanych przez administratora reguł:
   1. adres źródłowy
   2. adres docelowy
   3. port
   4. protokół
   5. kierunek
   6. akcja

## PB2: PRZEGLĄDANIE REGUŁ ZWIĄZANYCH ZE STEROWANIEM

Aktorzy: administrator

Scenariusz główny:

1. System sprawdza tożsamość i uprawnienia użytkownika
2. System wyświetla listę zdefiniowanych przez administratora reguł:
   1. adres docelowy
   2. funkcja
   3. rejestr
   4. akcja

# ANALIZA PRZYPADKÓW UŻYCIA: DODAWANIE REGUŁ

## PB3: DODAWANIE REGUŁY STANDARDOWEJ

Aktorzy: administrator

Scenariusz główny:

1. System sprawdza tożsamość i uprawnienia użytkownika.
2. Administrator tworzy nową regułę z następującymi parametrami:
   1. adres źródłowy
   2. adres docelowy
   3. port
   4. protokół
   5. kierunek
   6. akcja
3. Reguła zostaje przekazana na koniec listy reguł przez system.

Scenariusz alternatywny 1 - reguła istnieje:

1-2. Jak w scenariuszu głównym.

1. Reguła z podanymi parametrami już istnieje, więc jej kopia nie zostanie dodana.

Scenariusz alternatywny 2 - użytkownik nie ma uprawnień administratora:

1-2. Jak w scenariuszu głównym.

1. Reguła nie zostaje dodana, ponieważ użytkownik ma za małe uprawnienia.

## PB4: DODAWANIE REGUŁY ZWIĄZANEJ ZE STEROWANIEM

Aktorzy: administrator

Scenariusz główny:

1. System sprawdza tożsamość i uprawnienia użytkownika.
2. Administrator tworzy nową regułę z następującymi parametrami:
   1. adres docelowy
   2. funkcja
   3. rejestr
   4. akcja
3. Reguła zostaje przekazana na koniec listy reguł przez system.

Scenariusz alternatywny 1 - reguła istnieje:

1-2. Jak w scenariuszu głównym.

1. Reguła z podanymi parametrami już istnieje, więc jej kopia nie zostanie dodana.

Scenariusz alternatywny 2 - użytkownik nie ma uprawnień administratora:

1-2. Jak w scenariuszu głównym.

1. Reguła nie zostaje dodana, ponieważ użytkownik ma za małe uprawnienia.

# ANALIZA PRZYPADKÓW UŻYCIA: USUWANIE REGUŁ

## PB5: USUWANIE REGUŁY

Aktorzy: administrator

Scenariusz główny:

1. System sprawdza tożsamość i uprawnienia użytkownika
2. Administrator wybiera regułę, którą chce usunąć.
3. System usuwa regułę podaną regułę z listy.
4. Wszystkie reguły, które znajdowały się poniżej, zostają przesunięte o jedną pozycję w górę.

Scenariusz alternatywny - użytkownik nie ma uprawnień administratora:

1-2. Jak w scenariuszu głównym.

1. Reguła nie zostaje dodana, ponieważ użytkownik ma za małe uprawnienia.

# ANALIZA PRZYPADKÓW UŻYCIA: POZOSTAŁE AKCJE

## PB7: ANALIZA ZDARZEŃ BEZPIECZEŃSTWA

Aktorzy: administrator

Scenariusz główny:

1. System sprawdza tożsamość i uprawnienia użytkownika.
2. Administrator wybiera regułę, którą chce zmodyfikować i zmienia jej parametry:
   1. pozycję na liście
   2. parametry filtrowania

Scenariusz alternatywny - użytkownik nie ma uprawnień administratora:

1-2. Jak w scenariuszu głównym.

Reguła nie zostaje dodana, ponieważ użytkownik ma za małe uprawnienia.

## FU1: LOGOWANIE UŻYTKOWNIKA

Aktorzy: użytkownik, serwer WWW

Scenariusz główny:

1. Użytkownikowi zostaje wyświetlony formularz do zalogowania się do systemu.
2. Użytkownik wypełnia formularz i przesyła do weryfikacji.
3. Użytkownik uzyskuje dostęp do panelu administratora.

Scenariusz poboczny - użytkownik nie podał poprawnych danych:

1-2. Jak w scenariuszu głównym.

1. Użytkownik nie uzyskuje dostępu do panelu administratora.

Scenariusz poboczny - użytkownik nie jest administratorem.

1-2. Jak w scenariuszu głównym.

1. Użytkownik nie uzyskuje pełnego dostępu do panelu administratora - może jedynie przeglądać reguły.

## FU2: WYŚWIETLENIE AKTUALNEJ LISTY REGUŁ NA SERWERZE WWW

Aktorzy: administrator, serwer WWW

Scenariusz główny:

1. Administrator otwiera okno reguł na serwerze.
2. System pobiera listę reguł zapisanych w pliku konfiguracyjnym.
3. System wyświetla listę reguł administratorowi.

## FU3: MODYFIKOWANIE LISTY REGUŁ NA SERWERZE WWW

 Aktorzy: administrator, serwer WWW, firewall

Scenariusz główny:

1-3. Jak w FU2.

1. Administrator wybiera rodzaj reguły.
2. Administrator wprowadza modyfikacje listy reguł.
3. Zaktualizowana lista reguł zostaje zapisana w pliku.
4. Firewall zostaje poinformowany o zmianie przez serwer WWW.
5. Firewall ładuje zmiany.

## FU4: ANALIZA ZAPYTAŃ Z SERWERA PRZEZ FIREWALL

Aktorzy: firewall, komputer sterujący

Scenariusz główny:

1. Interfejs sieciowy firewalla przyjmuje pakiety sieciowe i umieszcza je w kolejce wejściowej.
2. Pakiety z kolejki wejściowej przekazywane są do analizy nagłówków pod kątem zgodności z regułami filtracji sieciowej.
3. Odebrane z serwera zapytanie następnie analizowane jest pod kątem zgodności z regułami filtracji zapytań do sterowania.
4. Przefiltrowane zapytanie jest wysyłane do komputera sterującego.

Scenariusz poboczny 1 - pakiet jest odrzucony na poziomie filtracji pakietów:

1-3. Jak w scenariuszu głównym

1. Pakiety zostają odrzucone.
2. Szczegóły zapisywane są do logów.
3. Do hosta źródłowego odsyłana jest informacja o odrzuceniu.

Scenariusz poboczny 2 - pakiet jest odrzucony na poziomie analizy zapytań:

1-4. Jak w scenariuszu głównym

1. Zapytanie zostają odrzucone.
2. Szczegóły zapisywane są do logów.
3. Do hosta źródłowego odsyłana jest informacja o odrzuceniu.

## FU5: ANALIZA ZDARZEŃ

Aktorzy: administrator, serwer WWW

Scenariusz główny:

1. Administrator wyznacza okres analizy zdarzeń.
2. System pobiera logi z wyznaczonego okresu.
3. Na podstawie listy reguł i logów tworzony jest widok historii zdarzeń.
4. Historia zdarzeń wyświetlona jest administratorowi.

Scenariusz poboczny 1 - pakiet jest odrzucony na poziomie filtracji pakietów

1-3. Jak w scenariuszu głównym

1. Pakiety zostają odrzucone.
2. Szczegóły zapisywane są do logów.
3. Do hosta źródłowego odsyłana jest informacja o odrzuceniu.