API SAOL-PA

Załącznik 1 - API SAOL v3.2

API SAOL-PA

Dokumentacja połączenia bezpośredniego punktów alarmowych
do centrali wojewódzkiej w systemie SAOL System.

System Alarmowania i Ostrzegania Ludności
Mazowiecki Urząd Wojewódzki w Warszawie
Plac Bankowy 3/5, 05-077 Warszawa
+48 22 692 64 71
kjanicki@mazowieckie.pl
www.mazowieckie.pl
Opracował: inż. Tomasz Marcinkowski
Wersja API SAOL-PA: 1.3

Spis Treści

1 API SAOL-PA MUW – w pigułce

API SAOL-PA jest narzędziem pozwalającym na komunikację centrali wojewódzkiej z punktami alarmowymi. API systemowe pozwala na wymianę asynchroniczną i synchroniczną pomiędzy SAOL MUW a Punktami Alarmowymi. W dokumencie znajduje się zbiór komend oraz sposób komunikacji w jaki należy komunikować się w systemie SAOL MUW. Zawarta w dokumencie specyfikacja pozwala na pełną integrację obcych Punktów Alarmowych z SAOL MUW.

2 Komunikacja – podstawowe informacje

System SAOL MUW widnieje pod publicznym adresem IP lub pod adresem sieciowym VPN. W dalszej części dokumentu będziemy posługiwać się skrótem „IP" , które będzie równe zapisowi „http://adres_ip_muw" lub „https://adres_ip_muw". Wszystkie funkcje API zostały stworzone w kontrolerze o nazwie *eth_api_inne.php i eth_api_sws_inne* , do którego należy odwoływać się w następujący sposób:

IP/eth_api_inne/nazwa_wykonywanej_funkcji lub IP/eth_api_sws_inne/nazwa_wykonywanej_funkcji

System SAOL MUW to aplikacja serwerowa, w dalszej części dokumentu SAOL MUW będzie określane jako „*serwer*", a punkt alarmowy jako „*klient*".

Wyróżniamy dwa sposoby uwierzytelnienia klienta:
- bez autoryzacji
- z autoryzacją

Dane do serwera przekazywane są za pomocą metody POST(urlc) na wskazany adres IP, kontroler i funkcję.

TCP jest protokołem działającym w trybie klient-serwer. Serwer oczekuje na nawiązanie połączenia na określonym porcie. Klient inicjuje połączenie do serwera. W protokole TCP do nawiązania połączenia pomiędzy dwoma hostami wykorzystywana jest procedura nazwana *three-way handshake*. W sytuacji normalnej jest ona rozpoczynana, gdy host A chce nawiązać połączenie z hostem B, procedura wygląda następująco:

- host A wysyła do hosta B segment SYN wraz z informacją o dolnej wartości numerów sekwencyjnych używanych do numerowania segmentów wysyłanych przez host A (np. 100) a następnie przechodzi w stan SYN-SENT,
- host B, po otrzymaniu segmentu SYN, przechodzi w stan SYN-RECEIVED i, jeżeli również chce nawiązać połączenie, wysyła hostowi A segment SYN z informacją o dolnej wartości numerów sekwencyjnych używanych do numerowania segmentów wysyłanych przez host B (np. 300) oraz segment ACK z polem numeru sekwencji ustawionym na wartość o jeden większą niż wartość pola sekwencji pierwszego segmentu SYN hosta A, czyli 101.
- host A, po odebraniu segmentów SYN i ACK od hosta B przechodzi w stan ESTABLISHED i wysyła do niego segment ACK potwierdzający odebranie segmentu SYN (numer sekwencji ustawiony na 301)
- host B odbiera segment ACK i przechodzi w stan ESTABLISHED
- host A może teraz rozpocząć przesyłanie danych

Używany format daty to czas uniksowy, czas POSIX – system reprezentacji czasu mierzący liczbę sekund od 1970 roku UTC, czyli od chwili zwanej początkiem epoki Uniksa (ang. *Unix Epoch*). Nie uwzględnia sekund przestępnych, zatem rzeczywista liczba sekund jakie upłynęły od początku epoki Uniksa jest większa o liczbę sekund przestępnych. W systemie operacyjnym Unix i pochodnych czas jest przedstawiany jako 32-bitowa liczba sekund, które upłynęły od 1 stycznia 1970. Daną tę interpretuje się jako liczbę ze znakiem (ang. *signed integer*), w której wartości ujemne nie są wykorzystywane, dlatego dostępny przedział czasu wynosi $2^{31}-1$ sekund, co daje wartość równą 2 147 483 647. Pierwsze 10^9 sekund od początku epoki Uniksa upłynęło 9 września 2001, godz. 01:46:40 GMT. Chwilę tę nazwano "Unix billennium". Systemy uniksowe były odporne na tzw. problem roku 2000: 32-bitowy Unix time wyczerpie się 19 stycznia 2038 o godz. 03:14:07 UTC – wtedy pojawi się problem roku 2038 . Obecnie trwają prace, które mają wyeliminować problem roku 2038. Niektóre strony internetowe umożliwiają śledzenie czasu uniksowego na bieżąco, zaś w aplikacji mobilnej

Google Play można ustawić czas uniksowy na stronę główną. W dalszej części dokumentu czas w formacie uniksowym będzie określany jako „*date*".

3 Uwierzytelnianie nowego klienta

3.1 Przekazanie danych do SAOL MUW

Pierwszy etap dodawania nowego klienta do serwera jest przekazanie adresu IP klienta do obsługi systemu SAOL MUW. Obsługa SAOL MUW wprowadza adres IP klienta do firewall'a, aby klient uzyskał możliwość jakiejkolwiek komunikacji z serwerem.
Obsługa SAOL MUW wygeneruje dla dedykowanego adresu IP klienta kod uwierzytelniający, który będzie niezbędny do prawidłowej komunikacji z SAOL MUW i przekaże go obsłudze klienta.
Do komunikacji między SAOL MUW a Punktem Alarmowym niezbędny jest klucz licencyjny, który Wykonawca systemu otrzymuje od MUW lub wykupuje klucz licencyjny u producenta lub dystrybutora systemu SAOL MUW i przekazuje obsłudze MUW.

3.2 Pierwszy pakiet danych - wysyłka

Po otrzymaniu kodu uwierzytelniającego od administracji SAOL MUW należy potwierdzić łączność wysyłając odpowiednią paczkę danych na adres:

IP/eth_api_inne/potwierdzam_uwierzytelnienie_klient

Należy przesłać dane zakodowane poprzez algorytm szyfrujący.
**Nazwa pola wysyłki – „session" -> dane: wygenerowany losowo unikalny identyfikator komendy składający się z alfa numerycznych znaków(wyłączając polskie znaki dialektyczne) o stałej długości 13 znaków.

Nazwa pola wysyłki - „date_send" -> dane: date (zakodowane logarytmem szyfrującym base64)

Nazwa pola wysyłki – „PIN" -> dane -> string utworzony z danych:

*date_send(kodowane base64), ***kod uwierzytelniający(string odwrócony – pisany od tyłu oraz kodowany base64),session
Podany string należy zakodować przed wysyłką metodą szyfrującą MD5 i następnie metodą szyfrującą SHA1.

Nazwa pola wysyłki – „check" -> dane: string składający się z pól:

**session,date_send,PIN*

Podany string należy zakodować przed wysyłką metodą szyfrującą SHA1 i następnie metodą szyfrującą MD5.

*String pisany jest w sposób ciągły, nie występują znaki interpunkcyjne pomiędzy seriami danymi, należy zachować kolejność przekazywanych danych oraz wielkości znaków
** W dalszej części dokumentu string składający się z alfa numerycznych znaków(wyłączając polskie znaki dialektyczne) będzie nazywany „session_id"
*** kod autoryzacyjny(string odwrócony – pisany od tyłu oraz kodowany base64) w dalszej części dokumentu będzie nazywany „OB_PIN"

3.3 Pierwszy pakiet danych - odbiór

Po poprawnym odebraniu danych przez serwer zostaną zwrócone dane na adres IP klienta:
Adres odpowiedzi:

```
IP_klient/eth_api/uwierzytelnienie
```

Nazwa pola odbioru – „session_id" -> dane: wygenerowany losowo unikalny identyfikator komendy składający się z alfa numerycznych znaków(wyłączając polskie znaki dialektyczne) o stałej długości 13 znaków.

Nazwa pola odbioru – „date_get" -> dane: date – data do synchronizacji czasu - serwer -> klient, klient -> serwer

Nazwa pola odbioru – „PIN" -> dane: string utworzony z danych:

* klucz_licencyjny,session_id ,date(kodowane base64), kod uwierzytelniający(string odwrócony – pisany od tyłu oraz kodowany base64)

Podany string jest zakodowany przed wysyłką metodą szyfrującą MD5 i następnie metodą szyfrującą SHA1.

Nazwa pola odbioru – „check" -> dane -> string składający się z pól:

* session_id,*date_send,PIN*

Podany string jest zakodowany przed wysyłką metodą szyfrującą SHA1 i następnie metodą szyfrującą MD5.

*String pisany jest w sposób ciągły, nie występują znaki interpunkcyjne pomiędzy seriami danymi

Nazwa pola wysyłki lub nazwa pola odbioru w dalszej części dokumenty będzie nazywany „*kluczem*".

3.4 Algorytm odbierania danych z serwera

Algorytm klienta odbierający dane z serwera musi sprawdzać poprawność otrzymywanych danych. Po stronie klienta należy sprawdzić:

1. Adres IP nadawcy - jeśli inny od adresu IP serwera dane powinny zostać natychmiast odrzucone, a adres IP, który próbował się komunikować zapisany do rejestru i wyświetlony obsłudze oprogramowaniu klienta.
2. Przyrównać wartość PINu odebranego, z PINem wygenerowanym z danych odebranych w pozostałych paczkach danych oraz z danymi zapisanymi tj. kod uwierzytelniający.
3. Sprawdzenie sumy kontrolnej, czy paczka danych nie uległa deformacji podczas przesyłania.

4 Pobieranie danych z Punktu Alarmowego

Dane do serwera przekazywane są za pomocą metody POST(urlc) na wskazany adres IP, kontroler i funkcję.

4.1 Pobieranie raportów urządzeń PA

Sól (ang. salt) lub ciąg zaburzający – dane losowe dodawane do PINu podczas obliczania funkcji skrótu przechowywanej w systemach informatycznych. Celem soli jest ochrona transmisji przesyłania danych przed atakami siłowymi, uniemożliwieniem ich odczytania, a zarazem możliwości wygenerowania nowej paczki danych na podstawie przechwyconych danych, w tym przed atakami typu man-in-the-middle. Jako że sól nie jest przechowywana jawnie, ma ona także znaczenie względem ataków brute-force.

```
IP/eth_api_sws_inne/urzadzenie_zadanie
```

Klucz – „session_id" -> dane: session_id
Klucz – „date" -> dane: date
Klucz – „identity" -> dane: identyfikator urządzenia
Klucz – „zadanie" -> dane: base64(SYRENA) lub base64(POGODA) lub base64(CZUJNIKS) lub base64(LIMNIMETR) lub base64(ZEGAR)
pisany od przodu(odebrany: 7654321ytrewą, wysłać: qwerty1234567)
Klucz – „PIN" -> dane -> string utworzony z danych: *klucz_licencyjny,OB_PIN, ,identity,date,session_id,zadanie*
Podany string należy zakodować przed wysyłką metodą szyfrującą SHA512. Wygenerowany string będzie długości 128 znaków.
String PIN dzielimy na dwie części:
Część pierwsza PINu – 26 pierwszych znaków stringa PIN po kodowaniu SHA512
Część druga PINu - wycięty string PINu po kodowaniu SHA512 od 26 znaku do 128 znaku

Następnie tworzymy PIN z ciągami zaburzającymi o długości 256 znaków:

Część pierwsza PINu + ciąg zaburzający 92znaki + Część druga PINu 102 znaki + ciąg zaburzający 36znaków

ŁĄCZNOŚĆ RADIOWA
Dane z tablicy asocjacyjnej należy zamienić na stringa (funkcja PHP serialize()) i wysłać przez łączność radiową zakańczając stringa znakiem końca wiersza (\r).

4.2 Zwracanie danych – SYRENA

W odpowiedzi na żądanie serwera, klient musi zwrócić string w następującej postaci:

OSYOCSXXN0010261130000000000000EWSK-K1 EL0261

bajt 01 do 06 OSYOCS - identyfikator odpowiedzi syreny
07 bajt X status syr XX
08 bajt X status syr (XX- syrena nic nie robi)
(A1- Włączony Alarm KATASTROFA ; K1- Włączony Komunikat Katastrofa ; U4 –Włączony kanał audio PA ; itd. Według konfiguracji centrali Alarmowej i sterownika syreny)
09 bajt - trening N- normalnie C- trening
10 bajt - zasilanie 24V 1- w normie 0- błędne
11 bajt - zasialnie 12V 1- w normie 0- błędne
12 bajt - drzwi syreny 0-drzwi zamknięte 1 – drzwi otwarte
13 bajt - zasilanie 230V 0- zasilanie 230V jest, 1- zasialni 230V brak
14 bajt do 16 - napiecie 24V odczyt 267 to znaczy 26.7V
17 bajt do 19 - napiecie 12V odczyt 132 to znaczy 13.2V
20 bajt - glosnik 1 - 1 lub 0 - sprawnosć głośników
21 bajt glosnik 2
22 bajt glosnik 3
23 bajt glosnik 4
24 bajt wzmacniacz 1 0 – wzmacniacz niesprawny 1- wzmacniacz sprawny
25 bajt wzmacniacz 2
26 bajt wzmacniacz 3
27 bajt wzmacniacz 4
28 bajt do 30 - napięcie 24V napięcie akumulatorów po teście głośników i wzmacniaczy odczyt 255 to znaczy 25.5V
31 bajt do 40 - nazwa syreny

41 bajt do 43 - prąd ładowania akumulatorów odczyt 035 to znaczy 0.35A
44 bajt - sprawność generatora odczyt 1 sprawny (0 - niesprawny)

4.3 Zwracanie danych – STACJA POGODOWA

W odpowiedzi na żądanie serwera, klient musi zwrócić string w następującej postaci:

OPG+261360400001003+261370400001003+261370400001003+26137040000100400000000

String zawiera cztery odpowiedzi stacji pogodowej z okresu ostatnich 40 minut, każdy pomiar wykonany jest z 10 minutowym opóźnieniem, pierwszy najstarszy, ostatni najnowszy. Najnowszy pomiar jest dłuższy od pozostałych danych wejściowych i zawiera informacje na temat opadów z ostatnich 24 godzin i bieżący stan opadów na moment pomiaru.

bajt 01 do 03 – identyfikator odpowiedzi stacji pogodowej.
bajt 04 – znak temperatury (+/-)
bajt 05 do 07 – temperatura, baj 07 to dziesiętna
bajt 08 do 09 – wilgotność
bajt 10 do 11 – kierunek wiatru
bajt 12 do 15 – prędkość wiatru
bajt 16 do 19 – ciśnienie

bajt 20 – znak temperatury (+/-)
bajt 21 do 23 – temperatura, baj 07 to dziesiętna
bajt 24 do 25 – wilgotność
bajt 26 do 27 – kierunek wiatru
bajt 28 do 31 – prędkość wiatru
bajt 32 do 35 – ciśnienie

bajt 36 – znak temperatury (+/-)
bajt 37 do 39 – temperatura, baj 07 to dziesiętna
bajt 40 do 41 – wilgotność
bajt 42 do 43 – kierunek wiatru
bajt 44 do 47 – prędkość wiatru
bajt 48 do 51 – ciśnienie

bajt 52 – znak temperatury (+/-)
bajt 53 do 55 – temperatura, baj 07 to dziesiętna
bajt 56 do 57 – wilgotność
bajt 58 do 59 – kierunek wiatru
bajt 60 do 63 – prędkość wiatru
bajt 64 do 67 – ciśnienie
bajt 68 do 71 – bieżące opady
bajt 72 do 75 – opady z 24 godzin

4.4 Zwracanie danych – CZUJNIK SKAŻEŃ

W odpowiedzi na żądanie serwera, klient musi zwrócić string w następującej postaci:

OIO00000011000000000401000014000016

OIO – identyfikator odpowiedzi czujnika skażeń

Bajt 26 do 28 – wartość średnia dawki promieniowania z ostatniej minuty
Bajt 32 do 34 – wartość średnia dawki promieniowania z ostatniej godziny

4.5 Zwracanie danych – LIMNIMETR

W odpowiedzi na żądanie serwera, klient musi zwrócić string w następującej postaci:

MIL00000010000000000000000000000034

MIL – identyfikator odpowiedzi limnimetru

Bajt 32 do 34 – napięcie z pętli prądowej urządzenia odczyt 034 mA

4.6 Zwracanie danych – ZEGAR DCF

W odpowiedzi na żądanie serwera, klient musi zwrócić string w następującej postaci:

ODC12005714031500010

ODC – identyfikator odpowiedzi Zegara DCF

bajt 4 do 5 – godzina
bajt 6 do 7 – minuta
bajt 8 do 9 – sekunda
bajt 10 do 11 – dzień
bajt 12 do 13 – miesiąc
bajt 14 do 15 – rok
bajt 19 – synchronizacja zegara DCF z zegarem atomowym Frankfurt nad Menem - 1 JEST, 0 BRAK

5 Uruchamianie Alarmów, Komunikatów i Komunikatów nadawanych przez mikrofon z Centrali Wojewódzkiej do Punktu Alarmowego

Uruchomienie alarmów i komunikatów w PA wykonuje się poprzez wysyłkę odpowiednich danych na adres:

IP/eth_api_sws_inne/alarm_on

Klucz – „session_id" -> dane: session_id
Klucz – „date" -> dane: date
Klucz – „identity" -> dane: identyfikator urządzenia
Klucz – „zdarzenie" -> identyfikator alarmu - unikalny string składający się z 13 znaków alfa numerycznych(z wyłączeniem polskich znaków dialektycznych)
Klucz – „PIN" -> dane -> string utworzony z danych: *klucz_licencyjny,OB_PIN, ,identity,date,session_id,zdarzenie*
Podany string należy zakodować przed wysyłką metodą szyfrującą SHA512. Wygenerowany string będzie długości 128 znaków.
String PIN dzielimy na dwie części:
Część pierwsza PINu – 42 pierwszych znaków stringa PIN po kodowaniu SHA512
Część druga PINu - wycięty string PINu po kodowaniu SHA512 od 42 znaku do 128 znaku

Następnie tworzymy PIN z ciągami zaburzającymi o długości 256 znaków:

Część pierwsza PINu + ciąg zaburzający 65znaki + Część druga PINu 86 znaki + ciąg zaburzający 63znaków

Klucz – „alarm" -> zadanie jakie zostanie uruchomione w puncie alarmowym
bajt 1 – A – Alarm ; K – Komunikat ; U – nadawanie przez mikrofon
bajt 2 – jaki alarm/komunikat (cyfra od 1 do 9) ; jeśli ogłaszanie przez mikrofon -> U
bajt 3 – Głośność od 2 do 9

Zatrzymanie nadawania alarmu/komunikatu, nadawania przez mikrofon w punkcie alarmowym:
Klucz – „alarm" -> STOP

ŁĄCZNOŚĆ RADIOWA
Dane z tablicy asocjacyjnej należy zamienić na stringa (funkcja PHP serialize()) i wysłać przez łączność radiową zakańczając stringa znakiem końca wiersza (\r).

5.1 Potwierdzenie odebrania komendy o włączeniu alarmu/komunikatu do systemów nadrzędnych

Punkt alarmowy po otrzymaniu zdarzenia włączenia alarmu/komunikatu odpowiada tymi samymi danymi na adres:

IP/eth_api_sws_inne/alarm_on_potwierdzenie

Klucz – „session_id" -> dane: session_id
Klucz – „date" -> dane: date
Klucz – „identity" -> dane: identyfikator urządzenia
Klucz – „zdarzenie" -> unikalny string składający się z 13 znaków alfa numerycznych(z wyłączeniem polskich znaków dialektycznych) / identyfikator włączonego alarmu/komunikatu otrzymanego od systemu nadrzędnego punkt 5 - Klucz -> zdarzenie

pisany od przodu(odebrany: 7654321ytrewą, wysłać: qwerty1234567)
Klucz – „PIN" -> dane -> string utworzony z danych: *klucz_licencyjny,OB_PIN,*
,identity,date,session_id,zdarzenie
Podany string należy zakodować przed wysyłką metodą szyfrującą SHA512. Wygenerowany string będzie długości 128 znaków.
String PIN dzielimy na dwie części:
Część pierwsza PINu – 42 pierwszych znaków stringa PIN po kodowaniu SHA512
Część druga PINu - wycięty string PINu po kodowaniu SHA512 od 42 znaku do 128 znaku

Następnie tworzymy PIN z ciągami zaburzającymi o długości 256 znaków:

Część pierwsza PINu + ciąg zaburzający 65znaki + Część druga PINu 86 znaki + ciąg zaburzający 63znaków

Klucz – „alarm" -> zadanie jakie zostanie uruchomione w puncie alarmowym
bajt 1 – A – Alarm ; K – Komunikat ; U – nadawanie przez mikrofon
bajt 2 – jaki alarm/komunikat (cyfra od 1 do 9) ; jeśli ogłaszanie przez mikrofon -> U
bajt 3 – Głośność od 2 do 9

Zatrzymanie nadawania alarmu/komunikatu, nadawania przez mikrofon w punkcie alarmowym:
Klucz – „alarm" -> STOP

ŁĄCZNOŚĆ RADIOWA
Dane z tablicy asocjacyjnej należy zamienić na stringa (funkcja PHP serialize()) i wysłać przez łączność radiową zakańczając stringa znakiem końca wiersza (\r).

6 Wysyłanie zdarzeń systemowych z PA do systemów nadrzędnych

Centrala Wojewódzka przyjmuje zdarzenie systemowe pod adresem:

IP/eth_api_inne/zdarzenie

6.1 Syrena wirnikowa

Klucz – „session_id" -> dane: session_id
Klucz – „date" -> dane: date
Klucz – „zadanie" -> dane: „IN-SW"
Klucz – „data_send" -> dane:
- identyfikator urządzenia
- status syreny: 1 – sprawna, 2 – niesprawna, 3 – brak łączności, 4 – ogłasza alarm(z uwzględnieniem ogłaszania alarmu np. 4A1; XX – spoczynek)
- pamięć alarmu : 0 – brak; 1 – sprawna syr. pamięć; 2- niesprawna syr. pamięć
- tryb pracy,(N – normalny, C - trening)
- zasilanie (brak, jest)
- informacja na temat przynależności do sektorów(zakres sektorów A B C D E F G H),

Z powyższych danych tworzymy string, a parametry rozdzielamy między sobą znakiem interpunkcyjnym „ , "(przecinek).
Klucz – „PIN" -> dane -> string utworzony z danych:
klucz_licencyjny,session_id,date,zadanie,data_send,OB_pin
Klucz – „check" -> dane -> string utworzony z danych: *session_id,date,zadanie,data_send,PIN*

ŁĄCZNOŚĆ RADIOWA
Dane z tablicy asocjacyjnej należy zamienić na stringa (funkcja PHP serialize()) i wysłać przez łączność radiową zakańczając stringa znakiem końca wiersza (\r).

6.2 Syrena elektroniczna

Klucz – „session_id" -> dane: session_id
Klucz – „date" -> dane: date
Klucz – „zadanie" -> dane: „IN-SEL"
Klucz – „data_send" -> dane:
- identyfikator urządzenia
- status syreny: 1 – sprawna, 2 – niesprawna, 3 – brak łączności, 4 – ogłasza komunikat/alarm(z uwzględnieniem jaki ogłasza alarm, lub jaki ogłasza komunikat np. 4A2 – ogłasza określony alarm, 4K1 – ogłasza określony komunikat, XX - spoczynek),
- pamięć alarmu : 0 – brak; 1 – sprawna syr. pamięć; 2- niesprawna syr. pamięć
- tryb pracy(N – normalny, C - trening),
- informacje na temat zasilania 24V(brak, w normie),
- informacje na temat zasilania 12V(brak, w normie),
- drzwi do syreny(otwarte, zamknięte),
- zasilanie 230V(brak, jest),
- napięcie 24 V z dokładnością do jednego miejsca po przecinku),
- napięcie 12 V z dokładnością do jednego miejsca po przecinku,
- sprawność głośników zestaw 1(sprawne, niesprawne),
- sprawność głośników zestaw 2(sprawne, niesprawne),
- sprawność wzmacniaczy zestaw 1(sprawne, niesprawne),
- sprawność wzmacniaczy zestaw 2(sprawne, niesprawne),
- napięcie akumulatora po teście z dokładnością do jednego miejsca po przecinku (V),
- prąd ładowania akumulatora (A),
- sprawność generatora(sprawny, niesprawny)

Z powyższych danych tworzymy string, a parametry rozdzielamy między sobą znakiem interpunkcyjnym „ , "(przecinek).

Klucz – „PIN" -> dane -> string utworzony z danych:
klucz_licencyjny,session_id,date,zadanie,data_send,OB_pin
Klucz – „check" -> dane -> string utworzony z danych: *session_id,date,zadanie,data_send,PIN*

6.3 Stacja pogodowa

Klucz – „session_id" -> dane: session_id
Klucz – „date" -> dane: date
Klucz – „zadanie" -> dane: „IN-PG"
Klucz – „data_send" -> dane:
- identyfikator urządzenia
- status stacji pogodowej: 1 – sprawna, 2 – niesprawna, 3 – brak łączności,
- data i godzina pomiaru(format *date*),
- temperatura w stopniach Celsjusza z dokładnością do jednego miejsca po przecinku,
- ciśnienie w hPa,
- wilgotność powietrza (%),
- siłę wiatru w m/s,
- kierunek wiatru,
- opady w mm/m2 z ostatniej godziny,
- opady w mm/m2 z ostatnich 24 godzin,

Z powyższych danych tworzymy string, a parametry rozdzielamy między sobą znakiem interpunkcyjnym „ , "(przecinek).

Klucz – „PIN" -> dane -> string utworzony z danych:
klucz_licencyjny,session_id,date,zadanie,data_send,OB_pin
Klucz – „check" -> dane -> string utworzony z danych: *session_id,date,zadanie,data_send,PIN*

6.4 Czujnik skażeń

Klucz – „session_id" -> dane: session_id
Klucz – „date" -> dane: date
Klucz – „zadanie" -> dane: „IN-CZG"
Klucz – „data_send" -> dane:
- identyfikator urządzenia
- status czujnika skażeń: 1 – sprawny, 2 – niesprawny, 3 – brak łączności,
- data i godzina pomiaru(format *date*),
- wartość średnia dawki promieniowania z ostatniej minuty w µSv/h
- wartość średnia dawki promieniowania z ostatniej godziny w µSv/h

Z powyższych danych tworzymy string, a parametry rozdzielamy między sobą znakiem interpunkcyjnym „ , "(przecinek).

Klucz – „PIN" -> dane -> string utworzony z danych:
klucz_licencyjny,session_id,date,zadanie,data_send,OB_pin
Klucz – „check" -> dane -> string utworzony z danych: *session_id,date,zadanie,data_send,PIN*

6.5 Limnimetr

Klucz – „session_id" -> dane: session_id
Klucz – „date" -> dane: date
Klucz – „zadanie" -> dane: „IN-CZW"
Klucz – „data_send" -> dane:
- identyfikator urządzenia
- status wodomierza: 1 – sprawny, 2 – niesprawny, 3 – brak łączności,
- data i godzina pomiaru(format *date*),
- pętla prądowa (mA)

Z powyższych danych tworzymy string, a parametry rozdzielamy między sobą znakiem interpunkcyjnym „ , "(przecinek).

Klucz – „PIN" -> dane -> string utworzony z danych:
klucz_licencyjny,session_id,date,zadanie,data_send,OB_pin
Klucz – „check" -> dane -> string utworzony z danych: *session_id,date,zadanie,data_send,PIN*

6.6 Alarmy

Klucz – „session_id" -> dane: session_id
Klucz – „date" -> dane: date
Klucz – „zadanie" -> dane: „IN-ALARM"
Klucz – „data_send" -> dane:
- jaki alarm (od A1 do A8)
- czy zostanie nadany komunikat po alarmie(T – tak, N – nie jeśli tak to jaki np. TK1)
- głośność(1 do 9),
- data (format *date*),
- na ilu syrenach podjęta próba uruchomienia
- identyfikatory syren na których została podjęta próba uruchomienia oddzielone separatorem „ <syr> "

Z powyższych danych tworzymy string, a parametry rozdzielamy między sobą znakiem interpunkcyjnym „ , "(przecinek).
Klucz – „PIN" -> dane -> string utworzony z danych:

klucz_licencyjny,session_id,date,zadanie,data_send,OB_pin, date
Klucz – „check" -> dane -> string utworzony z danych: *session_id,date,zadanie,data_send,PIN*

6.6.1 Alarm na pojedynczej syrenie

Klucz – „session_id" -> dane: session_id
Klucz – „date" -> dane: date
Klucz – „zadanie" -> dane: „IN-SYRALARM"
Klucz – „data_send" -> dane:
- jaki alarm (od A1 do A8)
- czy zostanie nadany komunikat po alarmie(T – tak, N – nie jeśli tak to jaki np. TK1)
- głośność(1 do 9),
- data (format *date*),
- identyfikator syreny

Z powyższych danych tworzymy string, a parametry rozdzielamy między sobą znakiem interpunkcyjnym „ , "(przecinek).

Klucz – „PIN" -> dane -> string utworzony z danych:
klucz_licencyjny,session_id,date,zadanie,data_send,OB_pin
Klucz – „check" -> dane -> string utworzony z danych: *session_id,date,zadanie,data_send,PIN*

6.6.2 Fizyczne uruchomienie alarmu z przycisku w syrenie

Klucz – „session_id" -> dane: session_id
Klucz – „date" -> dane: date
Klucz – „zadanie" -> dane: „MANUAL-SYRALARM"
Klucz – „data_send" -> dane:
- jaki alarm (od A1 do A8)
- czy zostanie nadany komunikat po alarmie(T – tak, N – nie jeśli tak to jaki np. TK1)
- głośność(1 do 9),
- data (format *date*),
- identyfikator syreny

Z powyższych danych tworzymy string, a parametry rozdzielamy między sobą znakiem interpunkcyjnym „ , "(przecinek).
Klucz – „PIN" -> dane -> string utworzony z danych:
klucz_licencyjny,session_id,date,zadanie,data_send,OB_pin
Klucz – „check" -> dane -> string utworzony z danych: *session_id,date,zadanie,data_send,PIN*

6.6.3 Potwierdzenie włączenia alarmu w syrenie

Klucz – „session_id" -> dane: session_id
Klucz – „date" -> dane: date
Klucz – „zadanie" -> dane: „AUTO-SYRALARM"
Klucz – „data_send" -> dane:
- potwierdzenie (1 – poprawnie, 0 – syrena uszkodzona)
- data (format *date*),
- identyfikator syreny

Z powyższych danych tworzymy string, a parametry rozdzielamy między sobą znakiem interpunkcyjnym „ , "(przecinek).
Klucz – „PIN" -> dane -> string utworzony z danych:
klucz_licencyjny,session_id,date,zadanie,data_send,OB_pin
Klucz – „check" -> dane -> string utworzony z danych: *session_id,date,zadanie,data_send,PIN*

6.7 Komunikaty

Klucz – „session_id" -> dane: session_id
Klucz – „date" -> dane: date

Klucz – „zadanie" -> dane: „IN-KOM"
Klucz – „data_send" -> dane:
- jaki komunikat (od K1 do K8) dla mikrofonu (UU)
- data (format *date*),
- na ilu syrenach podjęta próba uruchomienia
- identyfikatory syren na których została podjęta próba uruchomienia oddzielone separatorem „ <syr> "
Z powyższych danych tworzymy string, a parametry rozdzielamy między sobą znakiem interpunkcyjnym „ , "(przecinek).
Klucz – „PIN" -> dane -> string utworzony z danych:
klucz_licencyjny,session_id,date,zadanie,data_send,OB_pin
Klucz – „check" -> dane -> string utworzony z danych: *session_id,date,zadanie,data_send,PIN*

6.7.1 Komunikat na pojedynczej syrenie

Klucz – „session_id" -> dane: session_id
Klucz – „date" -> dane: date
Klucz – „zadanie" -> dane: „IN-SYRKOM"
Klucz – „data_send" -> dane:
- jaki alarm (od K1 do K8) dla mikrofonu (UU)
- data (format *date*),
- identyfikator syreny
Z powyższych danych tworzymy string, a parametry rozdzielamy między sobą znakiem interpunkcyjnym „ , "(przecinek).
Klucz – „PIN" -> dane -> string utworzony z danych:
klucz_licencyjny,session_id,date,zadanie,data_send,OB_pin
Klucz – „check" -> dane -> string utworzony z danych: *session_id,date,zadanie,data_send,PIN*

6.7.2 Fizyczne uruchomienie komunikatu z przycisku w syrenie

Klucz – „session_id" -> dane: session_id
Klucz – „date" -> dane: date
Klucz – „zadanie" -> dane: „MANUAL-SYRKOM"
Klucz – „data_send" -> dane:
- jaki alarm (od K1 do K8) dla mikrofonu (UU)
- data (format *date*),
- identyfikator syreny
Z powyższych danych tworzymy string, a parametry rozdzielamy między sobą znakiem interpunkcyjnym „ , "(przecinek).
Klucz – „PIN" -> dane -> string utworzony z danych:
klucz_licencyjny,session_id,date,zadanie,data_send,OB_pin
Klucz – „check" -> dane -> string utworzony z danych: *session_id,date,zadanie,data_send,PIN*

6.7.3 Potwierdzenie włączenia komunikatu w syrenie

Klucz – „session_id" -> dane: session_id
Klucz – „date" -> dane: date
Klucz – „zadanie" -> dane: „AUTO-SYRKOM"
Klucz – „data_send" -> dane:
- potwierdzenie (1 – poprawnie, 0 – syrena uszkodzona)
- data (format *date*),
- identyfikator syreny
Z powyższych danych tworzymy string, a parametry rozdzielamy między sobą znakiem interpunkcyjnym „ , "(przecinek).

Klucz – „PIN" -> dane -> string utworzony z danych:
klucz_licencyjny,session_id,date,zadanie,data_send,OB_pin
Klucz – „check" -> dane -> string utworzony z danych: *session_id,date,zadanie,data_send,PIN*

ŁĄCZNOŚĆ RADIOWA
Dane z tablicy asocjacyjnej (dot. Punkt 6 - wszystkie podpunkty) należy zamienić na stringa (funkcja PHP serialize()) i wysłać przez łączność radiową zakańczając stringa znakiem końca wiersza (\r).

6.8 Otwarcie/zamknięcie drzwi szafy sterowniczej

Klucz – „session_id" -> dane: session_id
Klucz – „date" -> dane: date
Klucz – „zadanie" -> dane: „DOOR-OPEN" – drzwi otwarte / „DOOR-CLOSED" – drzwi zamknięte
Klucz – „data_send" -> dane:
- data (format *date*),
- identyfikator syreny
Z powyższych danych tworzymy string, a parametry rozdzielamy między sobą znakiem interpunkcyjnym „ , "(przecinek).

Klucz – „PIN" -> dane -> string utworzony z danych:
klucz_licencyjny,session_id,date,zadanie,data_send,OB_pin
Klucz – „check" -> dane -> string utworzony z danych: *session_id,date,zadanie,data_send,PIN*

6.9 Zanik/powrót napięcia w punkcie alarmowym

Klucz – „session_id" -> dane: session_id
Klucz – „date" -> dane: date
Klucz – „zadanie" -> dane: „VOLTAGE-OK" – jest napięcie / „VOLTAGE-BAD" – brak napięcia
Klucz – „data_send" -> dane:
- data (format *date*),
- identyfikator syreny
Z powyższych danych tworzymy string, a parametry rozdzielamy między sobą znakiem interpunkcyjnym „ , "(przecinek).

Klucz – „PIN" -> dane -> string utworzony z danych:
klucz_licencyjny,session_id,date,zadanie,data_send,OB_pin
Klucz – „check" -> dane -> string utworzony z danych: *session_id,date,zadanie,data_send,PIN*

6.10 Inne zdarzenia gdzie indziej nie sklasyfikowane
Ze względu na ciągły rozwój oprogramowania SAOL MUW lista obsługiwanych zdarzeń stale rośnie. Obsługa zdarzeń innych jest analogiczna do schematów połączeń zawartych w rozdziale 6.