# Projekt i wdrożenie technik umożliwiających przechowywanie i gromadzenie w bazie LDAP informacji o usługach i zasobach POL-34/NASK

**Sprawozdanie z prac prowadzonych w ramach**
**Projektu LDAP**
**(pkt. 20-22 harmonogramu)**

Opracował zespół w składzie:
Andrzej Chrząszcz
Marcin Gołębiewski
Maciej Siciarek

NASK – sierpień 2003

**Spis treści:**

# 1 Zakres prac

W ramach prac zrealizowano wymienione poniżej punkty harmonogramu:

Pkt. 20. Projekt technik umożliwiających przechowywanie i gromadzenie w bazie LDAP informacji o usługach i zasobach POL-34/NASK: wymagania specyficzne w sieci NASK

Pkt. 21. Przygotowanie programów i skryptów automatyzujących umieszczanie w bazie LDAP informacji o usługach i zasobach POL-34/NASK oraz korzystanie z tej informacji: j.w.

Pkt. 22. Przygotowanie narzędzi i wdrożenie zasad korzystania z bazy LDAP do zbierania statystyk o zasobach centrów MAN i KDMO:

# 2 Założenia do prowadzonych prac

W ramach punktów 20 – 22 harmonogramu projektu LDAP, jako przedmiot badań i wdrożeń NASK, zostało wytypowane zagadnienie integracji istniejących i wdrażanych mechanizmów zbierania statystyk na styku sieci POL-34 oraz NASK z bazą danych opartą na protokole LDAP. Obecnie wykorzystywane rozwiązania pozwalają na monitorowanie i graficzną wizualizację wykorzystania zasobów, natomiast nie ma zaimplementowanych mechanizmów przechowywania dużej ilości statystyk dla ich przyszłego przetwarzania.

## 2.1 Styk POL-34 w Warszawie

Styk sieci POL-34 w Warszawie realizowany jest w oparciu o przełącznik szkieletowy ASX 1000 firmy Fore.
Urządzenie zapewnia ruch międzywęzłowy wewnątrz sieci POL-34 oraz podłączenie warszawskich użytkowników sieci POL-34. W chwili obecnej warszawski węzeł obsługuje łącza międzywęzłowe do Lublina, Łodzi i Białegostoku.
Na stykach klienckich podłączeni są: ATMAN, ICM, Internet Partners oraz NASK.

## 2.2 Obecnie zbierane statystyki

W chwili obecnej nie ma jednolitego systemu zbierającego statystyki i pozwalającego na spójne monitorowanie danych na wszystkich stykach urządzenia ASX-1000. Oddzielnie realizowane jest monitorowanie od strony przełącznika w sieci POL-34 oraz od strony użytkowników sieci, a w tym przez NASK.
Administratorzy sieci POL-34 używają do celów generowania statystyk aplikacji MRTG (Multi Router Traffic Grapher) pozwalającej na generowanie w formacie html, stron zawierających graficzną (*.jpg. *.png) wizualizację statystyk ruchowych na interfejsach wskazanych urządzeń.

### 2.3 Zapotrzebowanie na statystyki ruchowe oraz rola LDAP w procesie obsługi i zbierania danych o urządzeniach

Po przeprowadzeniu analizy potrzeb oraz funkcji systemu zbierania danych zdecydowaliśmy się na wykorzystanie serwera LDAP jako bazy plików tekstowych generowanych przez *rrdfetch* ( pkt. 3 w opisie działania systemu). Pozwala to na wykorzystanie serwera LDAP jako jednolitej platformy, wspólnej dla wszystkich użytkowników oraz administratorów, pozwalającej na monitorowanie ruchu i zbieranie dużej ilości statystyk, a w konsekwencji skalowanie przepustowości oraz wzajemne rozliczenia użytkowników i operatora.
Statystyki zbierane przez aplikacje typu MRTG, RRD TOOLS oraz pokrewne rozwiązania, mogą być na bieżąco wizualizowane przy pomocy zaimplementowanych w ww. aplikacjach mechanizmów a jednocześnie wygenerowane pliki zawierające porcje statystyk z określonego okresu czasu będą przechowywane w bazie LDAP dla dalszego przetwarzania przez systemy bilingowe.

## 3 Opis środowiska wdrożenia

Implementacji wdrożenia dokonano w środowisku składającym się z następujących elemntów:

- monitorowanego systemu – przełącznik ATM Catalyst 8500 na styku NASK/POL-34
- wdrożonego zestawu narzędzi RRD Tools
- skryptów PERL wykorzystujących mechanizmy aplikacji RRD Tools
- instalacji testowej serwera Open Ldap

Centralnym elementem systemu zbierania danych o zasobach sieciowych jest serwer Linux (Debian 3.0). Za pomocą zapytań SNMP zbiera on informacje o stanie interfejsów monitorowanych urządzeń. Informacje te są zapisywane na serwerze, a następnie przetwarzane przez zestaw programów o nazwie *rrd tools* ( Round Robin Database tools) wywoływanych w ramach autorskich skryptów perlowych. Efektem finalnym działania systemu są udostępniane w formacie HTML (generowane na żądanie) statystyki przepływu danych oraz ich graficzna reprezentacja w postaci wykresów.

## 4 Informacje o pakiecie Round Robin DataBase tools

Kluczową rolę w procesie pobierania i przetwarzania danych pozyskanych z urządzeń pełnią narzędzia z pakietu rrd tools. Główne polecenia wykorzystywane w skryptach perl to:

- **RRdcreate**
- **RRdupdate**
- **RRdgraph**
- **Rrdfetch**

Poniżej zamieszczamy informacje o przeznaczeniu i składni wywołania głównych narzędzi pakietu.

**rrdcreate**

**rrdtool create** *filename* [**--start**|**-b** *start_time*]
[**--step**|**-s** *step*] [**DS**:*ds-name*:*DST*:*heartbeat*:*min*:*max*] [**RRA**:*CF*:*xff*:*steps*:*rows*]

*filename*
Nazwa RRD, którą chcesz stworzyć. Powinna mieć rozszerzenie .rrd , jednak będzie działać z każdym innym.

**--start**|-b *start_time* (domyślnie: teraz – 10s)
Określa czas w sekundach od 1 stycznia 1970, kiedy mają zacząć gromadzić się dane. **rrdtool** nie przyjmie żadnej danej przed, ani w momencie określonym tą opcją.

**--step**|-s *step* (domyślnie: 300 sekund)
Określa podstawowy interwał w sekundach z jakim dane będą przekazywane do **RRD**.

**DS**:*ds-name*:*DST*:*heartbeat*:*min*:*max*
Pojedynczy **RRD** może przyjąć dane z kilku Źródeł Danych (DS). Na przykład: ruch wchodzący i wychodzący...
W opcjach DS musisz zdefiniować kilka podstawowych właściwości DS, którego chcesz użyć do wypełniania danymi **RRD**.

*ds-name* jest nazwą, przez którą będziesz się odwoływał do konkretnej DS z **RRD**.

*DST* określa Typ Źródła Danych. Musi być jednym z poniższych:
**GAUGE**
dla danych typu: temperatura w pomieszczeniu;
**COUNTER**
dla stale rosnących liczników. Zakłada się, że dane z niego pochodzące **nigdy** nie maleją, chyba że licznik „się przekręci". **rrdtool** bierze pod uwagę czy nastąpiło „przekręcenie się" licznika 32- czy 64-bitowego.
**DERIVE**
przechowuje pochodną ostatniej zmiany wartości licznika (np.: może mierzyć szybkość przybywania lub ubywania ludzi w pomieszczeniu). Opcja ta działa jak **counter**, jednak bez sprawdania „przekrecenia się" licznika.
**ABSOLUTE**
dla liczników, które resetują się podczas odczytywania. Jest to użyteczne przy szybkich licznikach z tendencją do „przekręcania się". Resetując licznik po odczytaniu uzyskujesz maksimum dostępnego czasu do nastepnego „przekrecenia się".

*heartbeat* definiuje maksymalny czas w sekundach miedzy dwoma danymi, po którym DS zostanie okreslony jako UNKNOWN

*min* i *max* są opcjonalnym określeniem zakresu liczbowego, w jakim powinny się zawierać dostarczane dane. Jeśli *min* i/lub *max* są określone, dane spoza wyznacznego przez nie zakresu są oznaczane UNKWOWN. Gdy nie zna się tego zakresu można ustawić obie zmienne na U.

**RRA**:*CF*:*xff*:*steps*:*rows*
Przeznaczeniem **RRD** jest przechowywanie danych w **RRA**. Archiwum zawiera liczbę danych ze wszystkich zdefiniowanych Źrodeł Danych i jest definiowane przez linię **RRA**.

Przychodząca dana jest najpierw dopasowywana do przedziału czasowego określonego w opcji **-s** stając się *Primary Data Point (PDP)*.

Jest też konsolidowana przy użyciu jednej z funkcji *CF*: **AVERAGE**, **MIN**, **MAX**, **LAST**.

*xff* definiuje jaka część przedziału konsolidacji może być oznaczona jako UNKNOWN, aby konsolidowana dana pozostała określona.

*steps* określa jak dużo *PDP* zostanie użytych do zbudowania *Consolidated Data Point (CDP)* który trafi do archiwum.

*rows* definiuje ilość wygenerowanych w powyższy sposób danych, przechowywanych w **RRA**

**rrdupdate**

**rrdtool update** *filename*
[**--template**|**-t** *ds-name*[:*ds-name*]...]
**N**|*timestamp*:*value*[:*value...*] [*timestamp*:*value*[:*value...*]...]

*filename*
Nazwa RRD, w którą chcesz wrzucić dane.

**--template**|**-t** *ds-name*[:*ds-name*]...
domyślnie, funkcja update oczekuje danych w tej samej kolejności jak zostało to zdefiniowane w **RRD**. Ta opcja pozwala na wyszczególnienie jakie źródła danych (DS) i w jakiej kolejności będą uaktualniane. Jeśli wybrana zmienna nie istnieje działanie uaktualnienia zostanie zakończone z błedem.

**N**|*timestamp*:value[:*value...*]
Dana użyta do uaktualnienia **RRD** uzyskana została w określonym czasie. Czas ten (data i godzina) może być podany jako sekundy od 1.1.1970 lub przez wpisanie litery „N" jako czas aktualny. Ujemne wartości czasu są odejmowane od czcasu aktualnego.

Brakującym elementem wywołania jest wartość aktualizowanego DS. Kolejność uaktualniania danych jest taka sama jak zostało to zdefiniowane w **RRA**. Jeśli nie ma danych dla konkretnego DS, można użyć litery **U**: `N:0.1:U:1`.

Format uzyskanych z DS wartości zależy od wybranego Typu DS (DST). Standardowo będzie on numeryczny, ale moduł zbierajacy dane może narzucić własny układ.

**rrdgraph**

**rrdtool graph** *filename* [**-s**|**--start** *seconds*] [**-e**|**--end** *seconds*] [**-x**|**--x-grid** *x-axis grid and label*]
[**-y**|**--y-grid** *y-axis grid and label*] [**--alt-y-grid**]
[**--alt-autoscale**] [**--alt-autoscale-max**]
[**--units-exponent**] *value*]> [**-v**|**--vertical-label** *text*]
[**-w**|**--width** *pixels*] [**-h**|**--height** *pixels*] [**-i**|**--inetrlanced**]
[**-f**|**--imginfo** *formatstring*] [**-a**|**--imgformat GIF**|**PNG**]
[**-z**|**--lazy**] [**-o**|**--logarithmic**] [**-u**|**--upper-limit** *value*]
[**-l**|**--lower-limit** *value*] [**-g**|**--no-legend**] [**-r**|**--rigrid**]
[**--step** *value*] [**-b**|**--base** *value*] [**-c**|**--color** *COLORTAG#rrggbb*] [**-t**|**--title** *title*] [**DEF:***vname=rrd:ds-name:CF*]
[**CDEF:***vname=rpn-expression*] [**PRINT:***vname*:*CF*:*format*]
[**GPRINT:***vname*:*CF*:*format*] [**COMMENT:***text*] [**HRULE:***value***#***rrggbb*[:*legend*]]
[**VRULE:***time***#***rrggbb*[:*legend*]]
[**LINE**{**1**|**2**|**3**}:*vname*[**#***rrggbb*[:*legend*]]] [**AREA:***vname*[**#***rrggbb*[:*legend*]]]
[**STACK:***vname*[**#***rrggbb*[:*legend*]]]

Funcja **graph** oprócz generowania wykresów, potrafi także wykonywać numeryczne raporty

*filename*
Nazwa wykresu, który chcesz wygenerować. Zalecane jest określenie typu pliku: `.gif` lub `.png`, mimo, że **rrdtool** nie wymusza tego. Jeżeli *filemane* jest określone jako „–" obraz będzie przesłany na STDOUT. Inne wyniki działania zostaną wstrzymane.

`PNG` jest rekomendowane ze względu na mniejszą o 40% wielkość pliku i o 20% krótszy czas generowania pliku, w porównaniu z `GIF`.
Jeśli nie zostanie wywołana żadna funkcja graficzna, wykres nie zostanie wygenerowany.

**-s**|**--start** *seconds* (domyślnie `end-1dzień`)

Czas kiedy wykres ma się rozpoczynać. Czas musi być podany w sekundach, wartość ujemna oznacza czas przed aktualną datą. Domyślnie generowany jest wykres obejmujący ostatni dzień.

**-e**|**--end** *seconds* (domyślnie `teraz`)
Czas kiedy wykres ma się zakończyć.

**-x**|**--x-grid** *x-axis grid and label* (domyślnie `autokonfiguracja`)
Oś X jest kłopotliwa do konfiguracji, więc jeśli nie masz specjalnych wymagań pozostaw to do automatycznego skonfigurowania.
Jeśli nie chcesz mieć siatki X-owej ustaw **none**.

Konfiguracja *x-axis grid and label* wymaga użycia formatu:
GTM:GST:MTM:MST:LTM:LST:LPR:LFM
Konfigurujesz trzy elementy: *G??* dla siatki podstawowej, *M??* dla siatki głównej i *L??* dla etykiet.
Najpierw podajesz miarę czasu *?TM*, następnie ich ilość *?ST*.
Dla etykiet podajesz jeszcze dokładność w sekundach *?PR* oraz format używany do generowania etykiet *?FM*.

Element *?TM* musi być jednym z listy:
**SECOND, MINUTE, HOUR, DAY, WEEK, MONTH, YEAR**

Na przykład chcąc mieć siatkę podstawową co 10 minut, główną co godzinę, z etykietami co godzinę powinieneś użyć następującej definicji:
`MINUTE:10:HOUR:1:HOUR:1:0:%X`

Dokładność ma wartość 0, ponieważ format `%X` jest ścisły. Jeśli etykietą ma być nazwa dnia, dokładność powinna mieć wartość 24 godzin (84600 sekund) by „Monday" pojawiło się na „środku dnia".

**-y**|**--y-grid** *grid step:label factor* (domyślnie `autokonfiguracja`)
Powoduje pojawienie się siatki pionowej z częstotliwością *grid step*. Częstotliwość głównej siatki, wraz z którą pojawiają się etykiety – wartości linii, wyznacza *label factor*.
Jeśli nie chcesz mieć siatki Y-owej ustaw **none**.

**--alt-y-grid**
Tworzy siatkę pionową zależną od skali osi Y. Linie siatki pojawiają się co 1, 2, 5 i 10 jednostek.

**--alt-autoscale**
Oblicza skalę wartości Y w zależności od minimalnej i maksymalnej wartości. Do stosowania przy małych wahaniach wartości (np.: 260 + 0,001*sin(x) )

**--alt-autoscale-max**
Jak wyżej, ale oblicza tylko maksymalną wartość. Minimalna, jeśli nie została inaczej zdefiniowana w linii komend, zostanie ustawiona na zero. Do stosowania przy włączonej kompresji danych na interfejsie, kiedy wynik może być większy niż szybkość linii.

**--unix-exponent** *value* (domyślnie `autokonfiguracja`)
Umożliwia skalowanie osi Y przy pomocy funkcji eksponencjalnej. Normalnie wykres jest skalowany jednostkami odpowiednimi do nadchodzących danych (k, M). Ta opcja umożliwia narzucenie odpowiadającego Ci skalowania miedzy $10^{-18}$ a $10^{18}$ (ale tylko –18, –15, ..., –3, 0, 3, ..., 15, 18).

**-v**|**--vertical-label** *text*
Pionowa etykieta np. lewej stronie wykresu, określająca użyte jednostki.

**-w**|**--width** *pixels* (domyślnie 400 pixeli)
Szerokość wykresu.

**-h**|**--wheight** *pixels* (domyślnie 100 pixeli)
Wysokość wykresu.

**-i**|**--interlanced** (domyślnie: `false`)

Włączając tę opcję, otrzymasz „interlaced” `GIF`. Przeglądarki pokazują takie pliki stopniowo, podczas wgrywania. *** Opcja dotyczy wyłącznie wielkości pliku na dysku.

**-f**|**--imginfo** *formatstring*
Po utworzeniu wykresu, funkcja **graph** używa `printf` wraz z powyższym formatowaniem, do wygenerowania wyniku odpowiadającego funkcji `PRINT`. Aby otrzymać etykietę **IMG** odpowiednią do dołączenia wykresu do strony WWW, komenda powinna mieć postać:
`--imginfo '<IMG SRC="/img/%s" WIDTH="%lu" HEIGHT="%lu" ALT="Demo">`

**-a**|**--imgformat GIF|PNG** (domyślnie: `GIF`)
Pozwala generować wykresy w formacie `PNG`.

**-z**|**--lazy** (domyślnie: `false`)
Generuje tylko wykres, jeśli taki nie istnieje lub jest nieaktualny.

**-u**|**--upper-limit** *value* (domyślnie `autokonfiguracja`)
Definiuje wartość umieszczaną przy górnej granicy wykresu. Jeśli wykres będzie zawierał większe wartości, górna granica zostanie przesunięta, by zmieścić także te wartości. Aby zdefiniować stałą górną granicę trzeba użyć opcji `--rigid`.

**-l**|**--lower-limit** *value* (domyślnie `autokonfiguracja`)
Nie jest to dolna granica wykresu. Jest to raczej rozszerzenie wykresu w dół.

**-r**|**--rigid**
Tryb sztywnych granic. Standardowo **rrdtool** rozszerza dolną i górną granicę wykresu, by umieścić na nim wszystkie wartości. Ta opcja wyłącza taką możliwość.

**-b**|**--base** *value*
Jeżeli wykres dotyczy pamięci, ten przełącznik powinien być ustawiony na 1024, jeśli ruchu w sieci – na 1000. Rozróżnia to Kb pamięci od kb/s ruchu:
`1 Kb = 1024 byte; 1 kb/s = 1000 b/s.`

**-o**|**--logarithmic**
Logarytmicznie skalowana oś Y.

**-c**|**--color** *COLORTAG***#***rrggbb* (domyślnie `colors`)
Zmienia kolory standardowych elementów wykresu. COLORTAG musi być jednym z następujących elementów: **BACK** tło, **CANVAS** kanwa, **SHADEA** lewy/górny brzeg, **SHADEB** prawy/dolny brzeg, **GRID** siatka, **MGRID** siatka główna, **FONT** czcionka, **FRAME** (*?*) i osie wykresu oraz **ARROW** strzałki. Opcja może być wywoływana wiele razy.

**-g**|**--no-legend**
Wstrzymuje generowanie legendy, tworzy tylko wykres.

**-t**|**--title** *text* (domyślnie bez tytułu)
Definiuje tytuł do dodania do wykresu.

**--step** *value* (domyślnie `automatic`)
Domyślnie **rrdtool** liczy szerokość jednego pixela na jeden okres czasu i z tą częstotliwością oczekuje danych z **RRD**. Ta opcja pozwala zmienić tą częstotliwość np.: na 1 godzinę (wtedy trzeba ustawić ją na 3600). Częstotliwość mniejsza niż 1 zostanie zignorowana.

**DEF:***vname=rrd:ds-name:CF*
Definiuje wirtualną nazwę źródła danych DS. Nazwa ta może być użyta w funkcjach opisanych poniżej. Wywołanie **DEF** automatycznie wybiera **RRA**, który zawiera dane konsolidowane funkcją *CF* z częstotliwością odpowiadającą wielkości tworzonego wykresu. W założeniach oznacza to, że jedna dana z **RRA** ma być reprezentowana na wykresie przez jeden pixel. Jeżeli częstotliwość **RRA** jest

większa niż częstotliwość wykresu, dane z **RRA** będą dodatkowo skonsolidowane stosownie do wybranej funkcji *CF*.

**CDEF:***vname=rpn-expression*
Tworzy nowe wirtualne źródło danych DS dokonując obliczeń określonych przy pomocy RPN (Odwrotnej Notacji Polskiej). Należy pamiętać, że można używać tylko takich *vname* które zostały już zdefiniowane we wcześniejszych wywołaniach **DEF** i **CDEF**.

**PRINT:***vname*:*CF*:*format*
Obliczenie wybranej funkcji konsolidacyjnej *CF* dla zmiennej *vname* i wydrukowanie wyników „printf" na **STDOUT** przy użyciu *formatu*. W ciągu *format* powinien być znacznik „%lf" lub „%le" w miejscu, gdzie ma się pojawić liczba.

Jeżeli po tym znaczniku pojawi się „%s", wartość zostanie wyskalowana i odpowiednia wartość SI zostanie wydrukowana w miejscu znacznika „%s". Skalowanie zostanie przeprowadzone przy zachowaniu opcji --base.

Jeżeli zostanie użyte „%S" zamiast „%s", użyte zostanie poprzednio obliczona wartość SI (zamiast liczyć ją od nowa). Jeżeli wcześnie nie była obliczana wartość SI, „%S" zachowa się jak „%s" chyba że wartość jest równa 0.

Jeśli chcesz umieścić „%" w tekście, użyj „%%".

**GPRINT:***vname*:*CF*:*format*
To samo co PRINT, ale drukowane pod legendą.

Uwaga: Kiedy używasz funkcji **PRINT** i **GPRINT** do przeliczania danych w przedziałach czasowych ograniczonych przez aktualny czas, pamiętaj że ostatnia próbka prawie na pewno będzie miała wartość UNKNOWN, jako znajdująca się poza ostatnią aktualizacja bazy danych. Jest to rezultat sposobu uzyskiwania danych, szczególnie w przypadku funkcji konsolidacyjnej **AVERAGE**. Aby usunąć ten efekt ustal koniec przedziału czasowego na wcześniejszy od aktualnego czasu o co najmniej jeden *heartbeat*.

**COMMENT:***text*
Tak jak **GPRINT**, ale tekst jest wypisywany wprost do wykresu (*?*).

**HRULE:***value*#*rrggbb*[:*legend*]
Drukuje na wykresie poziomy liniał, dodając opcjonalnie legendę.

**VRULE:***time*#*rrggbb*[:*legend*]
Drukuje na wykresie pionowy liniał, dodając opcjonalnie legendę.

**LINE**{**1**|**2**|**3**}:*vname*[#*rrggbb*[:*legend*]]
Drukuje żądane dane, używając wybranego koloru. Zapisuje legendę do tego wykresy. Linie mogą być 3 różnych grubości. Jeżeli nie został wybrany żaden kolor, jest rysowany niewidoczny wykres, co jest użyteczne z opcją **STACK** kiedy chcesz dodać wartości dwóch źródeł danych, bez pokazywania ich na wykresie.

**AREA:***vname*[#*rrggbb*[:*legend*]]
Wykonuje to samo co **LINE?**, ale pole między wartością 0, a wykresem zostanie wypełniona wybranym kolorem.

**STACK:***vname*[#*rrggbb*[:*legend*]]
Wykonuje to samo co **LINE?**, ale wykres będzie rysowany „nad" poprzednim wykresem **LINE**, **AREA** lub **STACK**. Zależnie od poprzedniego wykresu, **STACK** będzie albo typu **LINE**, albo typu **AREA**. To oznacza, że jako pierwszy musi być narysowany wykres typu **LINE**, albo **AREA**.

Należy pamiętać, że na wartością UNKWOWN **STACK** nic nie narysuje. Wyjściem z tej sytuacji jest zamiana danej UNKWOWN na 0 przy pomocy **CDEF**.

**rrddump**

**rrdtool dump** *filename.rrd > filename.xml*

**rrdfetch**

**rrdtool fetch** *filename CF* [-- **resolution**|-**r** resolution]
[-**start**|-**s** *start*] [--**end**|-**e** *end*]

Wydobywa dane z bazy **RRD**. Jest normalnie używane wewnątrz **graph**. **fetch** analizuje bazę danych i próbuje uzyskać dane w żądanym rozkładzie.

*filename*
Nazwa RRD, z której chcesz wydostać dane.

*CF*
Określa funkcję konsolidacyjną, która zostanie zastosowana do danych jakie chcesz uzyskać (AVERAGE, MIN, MAX, LAST)

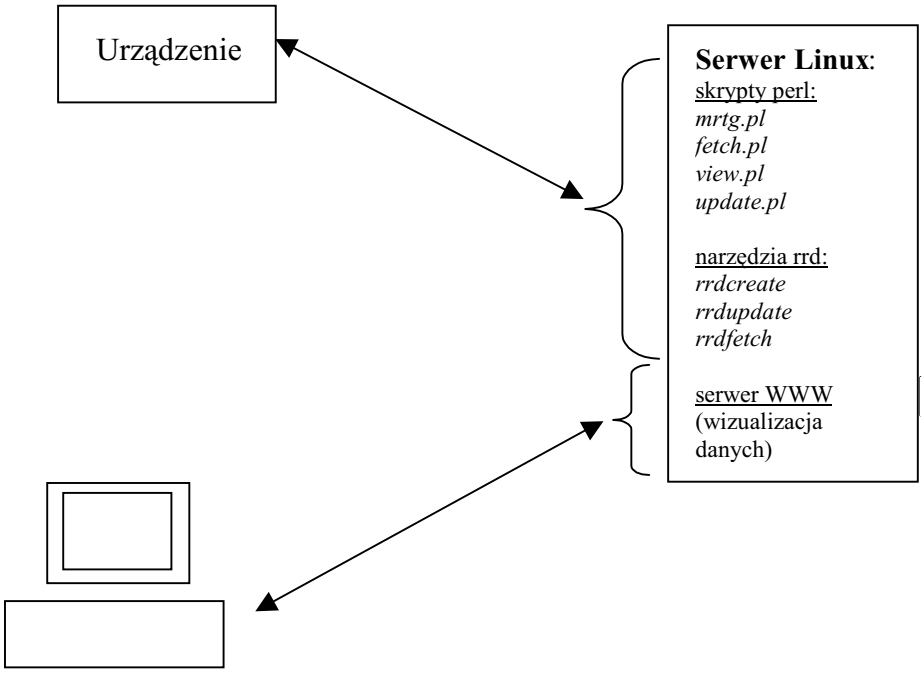
--**resolution**|-**r** *resolution* (domyślnie: najwyższa możliwa)
jaki interwał mają mieć dane (sekundy na wartość). **rrdfetch** będzie próbował dopasować dane, ale zwróci wartość, nawet jeśli całkowita zgodność nie będzie możliwa.

--**start**|-**s** *start* (domyślnie: end – 1 dzień)

--**end**|-**e** *end* (domyślnie: teraz)

**fetch** ma możliwość używania czasu w standardzie unixowej komendy AT

# 5 Model zbierania i udostępniania danych o stanie urządzeń bez integracji z LDAP-em




Stacja użytkownika
wyposażona w przeglądarkę

## 5.1 Opis etapów procesu zbierania i udostępniania danych o stanie urządzeń

1. Cyklicznie uruchamiany przez crontab (proces serwera Linux) skrypt o nazwie *update.pl* wykorzystując *rrdupdate* z pakietu *rrd tools* oraz protokół SNMP (funkcja *snmpget*) wykonuje uaktualnienie pliku (utworzonego po raz pierwszy za pomocą *rrdcreate*) przechowującego aktualne dane o stanie monitorowanego interfejsu urządzenia.

2. Uaktualniony plik (o nazwie *x_y.rrd* – gdzie *x* odpowiada urządzeniu, *y* jego interfejsowi) jest na żądanie użytkownika (kliknięcie odpowiedniego URL) prezentowany w przeglądarce stacji użytkownika. Za przetworzenie danych do formatu HTML odpowiedzialny jest skrypt *mrtg.pl* wykorzystujący *fetch.pl i rrdfetch* - dla wykonania różnicowej analizy pozyskanych danych oraz *view.pl* i *rrdgraph* dla wizualizacji danych (wykresy prezentowane są w formacie *jpg* i *png*).

3. W ramach procesu monitorowania stanu pracy urządzeń zachodzi także potrzeba dodatkowej, zewnętrznej analizy uaktualnień danych powstających w wyniku działania *fetch.pl* umieszczanych w *x_y.rrd.* W tym celu wynik działania *rrdfetch* poprzez unixowe przekierowanie strumienia *stdout* (znak polecenia >) jest zapisywany do przechowywanego na serwerze pliku tekstowego.

## 5.2 Programy realizujące proces zbierania danych

Poniżej zamieszczone są skrypty PERL realizujące opisany powyżej proces.

### Zapis w crontab uruchamiający skrypt *update.pl*

0,5,10,15,20,25,30,35,40,45,50,55 * * * * /usr/local/mrtg/rrd/bin/update.pl wawaIPB1.baz

### Skrypt *update.pl*

```
#!/usr/bin/perl -w
#################################################################
##                            UPDATE.PL                        #
##   updatowanie bazy RRD dla polaczenia/klienta na podstawie  #
##   informacji z pliku konfiguracyjnego *.baz                 #
#################################################################


my $plik = $ARGV[0];                                # baza podlaczen
my $bazaDIR = "/usr/local/mrtg/rrd/conf/";
my $rrdDIR = "/usr/local/mrtg/rrd/";
my $klient;
my @dane_kl;
my @aktual;


#############################################################################
#####    GLOWNY_poczatek

{
   $bazaDIR .= $plik;
   open (BAZA, $bazaDIR) ||
       die "nie mozna otworzyc BAZY: $!";
   while (defined ( $klient = <BAZA>) ) {
       chomp($klient);
       if ($klient !~ /^#/) {
           @dane_kl = rozbior($klient,"::");
            #if ($plik eq "krajIPK2.baz") {
            #    print "$dane_kl[0]\n";
            #}
           @aktual = odczyt_stanu(@dane_kl);
           $aktual[2] = $dane_kl[0];
           $aktual[3] = $dane_kl[7];
           $aktual[4] = $dane_kl[11];
           wrzuc_do_arc(@aktual);
      }
   }
   close (BAZA) || die "nie mozna zamknac BAZY: $!";
}

#####   GLOWNY_koniec
#############################################################################


####### Rozbior_rekordu
#  Rozklada na pojedyncze pola rekord dzielony znakiem
#  $rekord -- rekord z danymi
#  $znak   -- znak oddzielajacy pola
#######
sub rozbior {
    my $rekord = $_[0];
    my $znak   = $_[1];
    my @wyn;
    my $pom;
```

```
    @wyn = split(/$znak/,$rekord);
    $pom = skala($wyn[2]);                              # predkosc -> w b/s
    $wyn[2] = $pom;
    return @wyn;
}


####### Przeskalowuje_predkosc
#  Zmienia zapisz z 123k lub 123M na xyz b/s
#  $v -- wartosc w formacie XXk/M do przeskalowania
#########
sub skala {
    my $v = $_[0];
    my $pom;

    if ($v ne "?") {
        $pom = chop($v);
        $v = ($pom eq "M") ? $v * 1024000 : $v * 1000 ;
    }
    return $v;
}


####### Oczyt_stanu_licznikow
#  Odczytuje aktualny stan licznikow
#  @tab -- rozlozony rekord danych klienta
sub odczyt_stanu {
    # 0-klient,3-router,4-community,5-wersja,7-zbieranie,8-dana1,9-dana2
    my @tab = @_;
    my $in;
    my $out;

    if ($tab[7] == 1) {
        my $nr_snmp = `snmpget -Ovq -v $tab[5] $tab[3] $tab[4]
ipAdEntIfIndex.$tab[8]`;
         chomp($nr_snmp);
        $in  = `/usr/bin/snmpget -Ovq -v $tab[5] $tab[3] $tab[4]
ifInOctets.$nr_snmp`;
        $out = `/usr/bin/snmpget -Ovq -v $tab[5] $tab[3] $tab[4]
ifOutOctets.$nr_snmp`;
    } elsif ($tab[7] == 2) {
        $in  = `/usr/bin/snmpget -Ovq -v $tab[5] $tab[3] $tab[4]
ifInOctets.$tab[8]`;
        $out = `/usr/bin/snmpget -Ovq -v $tab[5] $tab[3] $tab[4]
ifOutOctets.$tab[8]`;
    } elsif ( ($tab[7] == 3) || ($tab[7] == 4) ) {
        $in  = `/usr/bin/snmpget -Ovq -v $tab[5] $tab[3] $tab[4] .$tab[8]`;
         $out = `/usr/bin/snmpget -Ovq -v $tab[5] $tab[3] $tab[4] .$tab[9]`;
    } elsif ($tab[7] == 5) {
        $in  = `/usr/bin/snmpget -Ovq -v $tab[5] $tab[3] $tab[4] .$tab[8]`
    } else {
        print STDERR "Nieokreslony typ zbierania w: UPDATE.PL! plik: $plik, klient:
$tab[0]\n";
    }
    chomp($in);
    if ($tab[7] == 5) {
        return ($in);
        #return ($tab[0],$tab[11],$tab[7],$in);
    } else {
        chomp($out);
        return ($in,$out);
        #return ($tab[0],$tab[11],$tab[7],$in,$out);
    }
}


####### Wrzucenie_danych_do_archiwum
```

```
#  Updatuje archiwum *.rrd danymi z snmpget
#  $. -- informacje i dane do update'u
#######
sub wrzuc_do_arc {
    my $in      = $_[0];
    my $out     = $_[1];
    my $klient  = $_[2];
    my $typ     = $_[3];
    my $katalog = $_[4];

    if ($typ == 5) {
        `/usr/bin/rrdtool update $rrdDIR$katalog$klient.rrd N:$in`;
    } else {
        `/usr/bin/rrdtool update $rrdDIR$katalog$klient.rrd N:$in:$out`;
    }
}
```

**Skrypt *mrtg.pl***

```
#!/usr/bin/perl -w
################################################################
#                           MRTG.PL                            #
#   program tworzacy liste klientow dla KLIENTA z wywolania,   #
#   w ktorej umieszcza wyliczone statystyki i wykresy ruchu#
#   $1 -- krotka_nazwa_klienta bez rozszerzenia                #
################################################################

use CGI qw(:standard :html2 :html3);
use POSIX qw(strftime);

$cgi = new CGI;

my $klient = $cgi->param("nazwa");                # klient
my $miara  = $cgi->param("miara");                # wybor jednostki "slowo"
my $zwrot  = $cgi->param("zwrot");                # wybor kierunku 0-ok, 1-zmiana

### ustawienia KATALOGOW bezwzglednych
my $bazaDIR = "/usr/local/mrtg/rrd/conf/";
my $progDIR = "/usr/local/mrtg/rrd/bin/";
my $rrdDIR  = "/usr/local/mrtg/rrd/";
my $viewDIR = "/rrd-bin/";                #albo rrd czyli www/rrd

my @dane_kl;                                      # do danych o KLIENCIE
my $czas;
my $tytul;
my $update; my $statsy;
my $term; my $wolaj;
my @kawalki;
my $pasmo;
my $kolor;
my $mail;
my $pelna;


################################################################
#####   GLOWNY_poczatek
{
    @dane_kl = rozbior(znajdz_kl($klient,$bazaDIR),"::");          # dane o KLIENT
    $pasmo = skala($dane_kl[2]);
    $rrdDIR .= $dane_kl[11];
    $tytul = $klient;
    $tytul =~ s/(\S)/\U$1/;
                        #for ( $i=0; $i<=$#dane_kl; $i++) {
                        #print "\t$i\t$dane_kl[$i]\n";
                        #}
                        #read(STDIN,$a,1);
                        #print "###\n";
    $czas = `rrdtool last $rrdDIR$klient.rrd`;             # ostatni update RRD
    chomp($czas);

    print header(-charset => 'ISO-8859-2');
    print start_html(-head => meta({-http_equiv => 'Refresh',
                                    -content    => '300'}),
                     -title => ("$tytul"),
                     -encoding => 'iso-8859-2',
                     -lang => 'pl');

    if ($dane_kl[7] > 4 || $miara eq "cells") {
        print h1("$dane_kl[1]");
    } elsif ($miara ne "bits") {
        die "bledna MIARA: $miara dla danych klienta: $dane_kl[0]";
    } else {
        print h1("$dane_kl[1] [$dane_kl[2]]");
    }
```

```
    print i(h3("$dane_kl[10]")),hr;

    $update = ostatniUpdate($czas);
    print $update,hr;

    if ($dane_kl[0] !~ /^cpu/ && $dane_kl[11] !~ /klienci/) {
        kierunek(\@dane_kl);
        print hr;
    }


    $pelna = $dane_kl[1];
    $pelna = zmien_space($pelna);
    $wolaj = konkatenacja($viewDIR,"analysis1.pl");
    $statsy  = "$wolaj?nazwa=$dane_kl[0]&miara=$miara&zwrot=$zwrot&pelna=$pelna";
    print "<A HREF=\"$statsy\" target=\"_blank\">Analiza ruchu</A>";
    print "     <BR>",hr;
    #$wolaj = konkatenacja($viewDIR,"zakres.pl");
    #my $statki  =
"$wolaj?nazwa=$dane_kl[0]&miara=$miara&zwrot=$zwrot&pelna=$pelna";
    #print "<A HREF=\"$statki\" target=\"_blank\">Zmiana okresu statystyk</A>",hr;


#foreach my $term ('day') {
foreach my $term ('day','week','month','year') {
        if ($term eq 'day') {
            print b('Wykres "Dzienny" (średnia 5-minutowa)'),p;
        } elsif ($term eq 'week') {
            print b('Wykres "Tygodniowy" (średnia 30-minutowa)'),p;
        } elsif ($term eq 'month') {
            print b('Wykres "Miesięczny" (średnia 2-godzinna)'),p;
        } else {
            print b('Wykres "Roczny" (średnia 1-dniowa)'),p;
        }

        $wolaj = konkatenacja($progDIR,"fetch.pl");
                                                # przeliczona predkosc
        $statsy = `$wolaj $dane_kl[0] $pasmo $term $czas $rrdDIR $dane_kl[7] $miara
$zwrot`;
        @kawalki = podzielTXT($statsy,";;");

        $wolaj = konkatenacja($viewDIR,"view1.pl");
        print "<img
src=\"$wolaj?nazwa=$dane_kl[0]\&predkosc=$pasmo\&max=$kawalki[0]\&okres=$term\&czas
=$czas\&katalog=$rrdDIR\&typ_zb=$dane_kl[7]\&miara=$miara\&zwrot=$zwrot\">",p;
        #print STDERR "<img
src=\"$wolaj?nazwa=$dane_kl[0]\&predkosc=$pasmo\&max=$kawalki[0]\&okres=$term\&czas
=$czas\&katalog=$rrdDIR\&typ_zb=$dane_kl[7]\&miara=$miara\&zwrot=$zwrot\">",p;
        print $kawalki[1],hr;
        #print STDERR "#$kawalki[0]#\n";
        #print $kawalki[0],$kawalki[1],hr;
    }
    $kolor = legendaKOLOR($miara);
    print br,h4("$kolor"),hr;
    $mail = mailing();
    print br,p("$mail"),hr;
    print end_html();

}
#####   GLOWNY_koniec
################################################################


####### Mailing
#  Generuje maila do administratora RRD
#######
sub mailing {
    my $wyn;
```

```
    $wyn  = "<H4><A HREF=\"mailto:bill\@nask.pl?subject=Problemy w RRDtool dla
$klient\">";
    $wyn .= "Poinformuj administratora</A></H4>";
    return $wyn;
}


####### Legenda_Kolor
#  Generuje legende kolorow
#######
sub legendaKOLOR {
    my $wyn;
    my $leg;

    if ($_[0] eq "bits" || $_[0] eq "cells") {
        $leg = $_[0];
         $leg =~ s/(\S)/\U$1/;
         $leg .= " per Second";

        $wyn  = "<TABLE WIDTH=500 BORDER=0 CELLPADDING=4 CELLSPACING=0>\n";
        $wyn .= "<WIDTH=500 BORDER=0 CELLPADDING=4 CELLSPACING=0>\n";
         $wyn .= "<TR><TD ALIGN=RIGHT><FONT SIZE=-1 COLOR=\"#00cc00\">";
         $wyn .= "<B>GREEN ###</B>";
         $wyn .= "</FONT></TD>\n<TD><FONT SIZE=-1>Incoming Traffic in $leg</FONT>";
         $wyn .= "</TD></TR>\n<TR><TD ALIGN=RIGHT><FONT SIZE=-1 COLOR=\"#0000ff\">";
         $wyn .= "<B>BLUE ###</B></FONT></TD>\n";
         $wyn .= "<TD><FONT SIZE=-1>Outcoming Traffic in $leg</FONT></TD></TR>\n";
        $wyn .= "<TR><TD ALIGN=RIGHT><FONT SIZE=-1 COLOR=\"#006600\">";
         $wyn .= "<B>DARK GREEN ###</B></FONT></TD>\n";
        $wyn .= "<TD><FONT SIZE=-1>Maximal 5 Minute Incoming Traffic</FONT>";
         $wyn .= "</TD></TR>\n<TR><TD ALIGN=RIGHT><FONT SIZE=-1 COLOR=\"#ff00ff\">";
         $wyn .= "<B>MAGENTA ###</B></FONT></TD>\n";
        $wyn .= "<TD><FONT SIZE=-1>Maximal 5 Minute Outgoing Traffic</FONT>";
         $wyn .= "</TD></TR></TABLE>\n";
    } else {
        $leg = $_[0];
         $leg =~ s/(\S)/\U$1/;

         $wyn  = "<TABLE WIDTH=500 BORDER=0 CELLPADDING=4 CELLSPACING=0>\n";
        $wyn .= "<WIDTH=500 BORDER=0 CELLPADDING=4 CELLSPACING=0>\n";
         $wyn .= "<TR><TD ALIGN=RIGHT><FONT SIZE=-1 COLOR=\"#00cc00\">";
         $wyn .= "<B>GREEN ###</B>";
         $wyn .= "</FONT></TD>\n<TD><FONT SIZE=-1>Level of $leg in time</FONT>";
         $wyn .= "<TR><TD ALIGN=RIGHT><FONT SIZE=-1 COLOR=\"#006600\">";
         $wyn .= "<B>DARK GREEN###</B></FONT></TD>\n";
         $wyn .= "<TD><FONT SIZE=-1>Maximal 5 Minute
Level</FONT></TD></TR></TABLE>\n";
    }
    return $wyn;
}


####### Znajdz_klienta
#  Znajduje w odpowiedniej bazie rekord z danymi o kliencie
#  $cli -- KLIENT z wywolania programu
#  $dirname -- katalog z bazami *.BAZ
#######
sub znajdz_kl {
    my $cli     = $_[0];
    my $dirname = $_[1];

    my $file; my $link;
    my @tab; my @wyn;

    opendir (DIR,$dirname)   || die "nie moge otworzyc katalogu z bazami: $dirname:
$!";
    while ( defined($file = readdir(DIR))) {
```

```
        $link = $dirname;
        if ($file =~ /\.baz/) {
            $link .= $file;
            open (BAZ, $link)   || die "nie moge otworzyc bazy $file: $!";
             @tab = <BAZ>;
            close (BAZ);
            @wyn = grep {/^$cli\:\:/} @tab;
             last if (@wyn);
         }
    }
    closedir (DIR);
    if (@wyn) {
        return @wyn;
    } else {
        die "nie moge znalezc klienta $cli!";
    }
}


####### Rozbior_rekordu
#  Rozklada na pojedyncze pola rekord dzielony znakiem
#  $rekord -- rekord z danymi
#  $znak   -- znak oddzielajacy pola
#######
sub rozbior {
    my $rekord = $_[0];
    my $znak   = $_[1];
    my @wyn;

    @wyn = split(/$znak/,$rekord);
    return @wyn;
}


####### Zamiana_bitow_na_cellki
#  Zamiana predkosci w bitach na cellki
#  $_[0] -- predkosc w bitach
#######
sub bit_cell {
    my $pom;

    if ($_[0] ne "?") {
        $pom = ($_[0] - ($_[0] % 53)) / 53;
    }
    return $pom;
}


######## Kierunek_na_wykresie
#  Ustala kierunek na wykresie
#  $p -- dane klienta
########
sub kierunek {
    my $p = $_[0];
    my @n;

    @n = split (/ /, ${$p}[1]);
    if ($zwrot) {
        print "<TABLE ALIGN=CENTER NOBORDER WIDTH=100%> <TR> <TD WIDTH=5%>\n";
         print "</TD> <TD WIDTH=60%>\n<FONT COLOR=\"RED\">\n";
         print "<B>UWAGA: zamienione kierunki na wykresach!\n</B></FONT>";
         print "</TD> <TD WIDTH=30%>\n</TD> </TR> </TABLE>\n";
        print "<TABLE ALIGN=CENTER NOBORDER WIDTH=100%> <TR> <TD WIDTH=5%>\n";
         print "</TD> <TD WIDTH=30%>\n<FONT COLOR=\"#00cc00\"><B>_IN_</FONT>\n";
         print "do</B> $n[3]<BR>\n";
         print "</TD> <TD WIDTH=5%>\n</TD> <TD WIDTH=30%>\n";
         print "<FONT COLOR=\"#0000ff\"><B>_OUT_</FONT>\n";
         print "od</B> $n[3]\n";
```

```
        print "</TD> <TD WIDTH=40%>\n</TD> </TR> </TABLE>\n";
    } else {
        print "<TABLE ALIGN=CENTER NOBORDER WIDTH=100%> <TR> <TD WIDTH=5%>\n";
        print "</TD> <TD WIDTH=30%>\n<FONT COLOR=\"#00cc00\"><B>_IN_</FONT>\n";
        print "do</B> $n[0]<BR>\n";
        print "</TD> <TD WIDTH=5%>\n</TD> <TD WIDTH=30%>\n";
        print "<FONT COLOR=\"#0000ff\"><B>_OUT_</FONT>\n";
        print "od</B> $n[0]<BR>";
        print "</TD> <TD WIDTH=40%>\n</TD> </TR> </TABLE>\n";
    }
}


####### Ostatni_update
#  Okresla czas ostaniego update'u RRD klienta
#  $klient -- klient
#  $czas -- aktualny czas
#######
sub ostatniUpdate {
    my $t = $_[0];
    my $pom = strftime " %A, %e of %B %Y, at %H:%M:%S", localtime($t);
    my $wyn;

        # wiecej niz 2h bez nowych danych
    if ($t < (time() - 7200)) {
        #print "\nThe statistics were last updated on <B>", $pom,
        $wyn = "\nThe statistics were last updated on <B> $pom <BR> <FONT
COLOR=\"#00cc00\"><H2> STATYSYKI SA NIEAKTUALNE! </H2></FONT> </B> <BR> \n";
    } else {
        $wyn = "\nThe statistics were last updated on <B> $pom </B>\n";
        #print "\nThe statistics were last updated on <B>", $pom, "</B>\n";
    }
    return $wyn;
}


####### Zmien_space_na_%20
#  Zmienia SPACE na hexadecymalne "%20"
#  $text -- tekst ze spacjami
#######
sub zmien_space {
    my $text = $_[0];
    my @a;
    my $i;

    @a = split(//,$text);
    for ($i = 0; $i <= $#a; $i++) {
        if ($a[$i] eq " ") {
            $a[$i] = "%20";
        }
    }
    $text = join("",@a);
    return $text;

}


####### Konkatenuje_napisy
#  Laczy w jeden napis-wywolanie dwa napisy: katalog i nazwe skryptu
#  $dir -- katalog
#  $pl -- skrypt
#######
sub konkatenacja {
    my $dir = $_[0];
    my $pl  = $_[1];

    $dir .= $pl;
    return $dir;
```

```
}


####### Podziel_tekst
#  Dzieli tekst na pola wedlug zadanego separatora
#  $txt -- tekst do podzialu
#  $ch  -- separator
#######
sub podzielTXT
{
    my $txt = $_[0];
    my $ch  = $_[1];
    my @tab;

    @tab = split(/$ch/,$txt);
    return @tab;
}


####### Przeskalowuje_predkosc
#  Zmienia zapis z 123k lub 123M na xyz b/s
#  $v -- wartosc w formacie XXk/M do przeskalowania
#######
sub skala {
    my $v = $_[0];
    my $pom;

    if ($v ne "?") {
        $pom = chop($v);
        $v = ($pom eq "M") ? $v * 1024000 : $v * 1000 ;
    }
    return $v;
}

# a=ą A=Ą c=ć C=Ć e=ę E=Ę l=ł L=Ł n=ń N=Ń o=ó O=Ó s=ś S=Ś z=ź Z=Ź z*=ż Z*=Ż
```

**Skrypt *fetch.pl***

```
#!/usr/bin/perl -w
################################################################
#                              FETCH.PL                        #
#   wylicznie SREDNICH i ich WYDRUK        wersja HTML        #
#   $1 -- krotka_nazwa_klienta bez sciezki/zrozszerzenia   #
#   $2 -- predkosc klienta w b/s                              #
#   $3 -- okres rozliczeniowy { day/week/month/year }         #
#   $4 -- "aktualny" czas bazy - ostatnie uaktualnienie RRD#
#   $5 -- katalog rrd                                         #
#   $6 -- typ zbierania danych                                #
#   $7 -- miara danych                                        #
#   $8 -- zwrot danych                                        #
################################################################


use POSIX qw(strftime);
use Time::Local;
use Date::Manip qw(ParseDate UnixDate);

my $nazwa  = $ARGV[0];                    # nazwa klienta
my $speed  = $ARGV[1];                    # predkosc klienta w
jednostakach_per_second
my $okres  = $ARGV[2];                    # okres wykresu
my $czas   = $ARGV[3];                    # aktualny czas
my $rrdDIR = $ARGV[4];                    # bezwzgledny katalog RRD
my $typ_zb = $ARGV[5];                    # typ zbieranych danych
bajty/cellki/inne_wartosci
my $miara  = $ARGV[6];                    # wybor jednostek:
bits/cells/percent/degris
my $zwrot  = $ARGV[7];                    # wybor zwrotu danych: 0-ok, 1-odwoc

my $h1;                                   # poczatek przedzialu czasowego
my $h2;                                   # koniec przedzialu czasowego
my %wyniki;                               # statystyki

#my $odej = 11 * 86400;
#$czas -= $odej;


################################################
#####   GLOWNY_poczatek
{
   my @s;
   my @t;
   my $r;

   if ($okres eq 'day') {
       $h2 = $czas - ($czas % 300);
       $h1 = $h2 - 150000;
       $r = "-r 300";
   } elsif ($okres eq 'week') {
       $h2 = $czas - ($czas % 1800);
       $h1 = $h2 - 900000;
       $r = "-r 1800";
   } elsif ($okres eq 'month') {
       $h2 = $czas - ($czas % 7200);
       $h1 = $h2 - 3600000;
       $r = "-r 7200";
   } else {
       $h2 = $czas - ($czas % 86400);
       $h1 = $h2 - 43200000;
       $r = "-r 86400";
   }
   #print STDERR "$okres\nrrdtool fetch $rrdDIR$nazwa.rrd $r -s $h1 -e $h2\n";

   if ($okres eq 'day') {
```

```
       @s = `rrdtool fetch $rrdDIR$nazwa.rrd AVERAGE $r -s $h1 -e $h2`;
       my $aaa = $#s;
       %wyniki = stats( tablica(\@s,$zwrot,$typ_zb),$typ_zb );              #
obliczenie SREDNICH
   } else {
       @s = `rrdtool fetch $rrdDIR$nazwa.rrd AVERAGE $r -s $h1 -e $h2`;
       @t = `rrdtool fetch $rrdDIR$nazwa.rrd MAX $r -s $h1 -e $h2`;
       %wyniki = stats(
tablica(\@s,$zwrot,$typ_zb),$typ_zb,tablica(\@t,$zwrot,$typ_zb) );
   }
   #mojprthash(\%wyniki);
   wydruk(\%wyniki,$speed,$typ_zb,$miara);                  # generownie wykresu
HTML

}
#####   GLOWNY_koniec
################################################


####### Drukownie_hasha
#  Drukuje hasha referencji do tablic
#  %hash -- hash referencji
#######
sub mojprthash {
    $hash = $_[0];

    foreach my $e (keys(%$hash)) {
        print STDERR "\n";
        print STDERR "$e: ";
        foreach my $f ( keys ( %{ ${%$hash}{$e}})) {
            print STDERR "$f -> ${%$hash}{$e}{$f} ";
        }
        print STDERR "\n";
    }
}


####### Wypelnienie_hasha_danymi
#  Wrzuca dane do hasha referencji do tablic
#  $refTAB -- wyjscie z rrdtool fetch
#  $kier   -- zwrot danych
#  $typ    -- typ zbierania
#######
sub tablica {
    my $refTAB = $_[0];
    my $kier   = $_[1];
    my $typ    = $_[2];
    my %hash;
    my @pole;

    shift(@{ $refTAB});
    shift(@{ $refTAB});
    my ($i, $match_idx);
    if ($typ == 5) {
        for ($i = 0; $i < @{ $refTAB}; $i++) {
            @pole = split(/ /,${ $refTAB}[$i]);
            chomp($pole[1]);
            if ($pole[1] ne "nan") {
                $match_idx = $i;
                last;
            }
         }
    } else {
        for ($i = 0; $i < @{ $refTAB}; $i++) {
            @pole = split(/ /,${ $refTAB}[$i]);
            chomp($pole[1]);
            chomp($pole[2]);
            if ($pole[1] ne "nan" || $pole[2] ne "nan") {
```

```
                $match_idx = $i;
                 last;
            }
         }
    }
    if (defined $match_idx) {
        for ($i = $match_idx; $i < @{ $refTAB}; $i++) {
            @pole = split(/ /,${ $refTAB}[$i]);
            chop($pole[0]);
            if ($typ == 5) {
                chomp($pole[1]);
                push( @{ $hash{'time'} }, $pole[0]);
                push( @{ $hash{'count'} }, (($pole[1] eq "nan") ? 0 : $pole[1]) );
            } else {
                chomp($pole[1]);
                chomp($pole[2]);
                push( @{ $hash{'time'} }, $pole[0]);
                if ($kier) {                                # PRAWDA = zamiana
zmiennych
                    push( @{ $hash{'in'} }, (($pole[2] eq "nan") ? 0 : $pole[2]) );
                    push( @{ $hash{'out'} }, (($pole[1] eq "nan") ? 0 : $pole[1])
);
                } else {
                    push( @{ $hash{'in'} }, (($pole[1] eq "nan") ? 0 : $pole[1]) );
                    push( @{ $hash{'out'} }, (($pole[2] eq "nan") ? 0 : $pole[2])
);
                }
            }
        }
    } else {
        print STDERR "FETCH: dla zakresu=$okres otrzymalem z archiwum dane NaN\n";
    }
    return {%hash};
}


####### Maximum_Srednia_Ostatni
#  Oblicza maxima, srednie i znajduje ostatnia dana z tablicy
#  @tab -- tablica danych do oblicznia
#######
sub max_avg_last {
    my @tab = @_;
    my $max;
    my $q;

    if (defined($tab[0])) {
        $max = $tab[0]
    } else {
        return;
    }
    my $sum = 0;
    foreach my $i (@tab) {
         if ($max < $i) {
             $max = $i };
         $sum += $i;
    }
    my $avg = $sum / ($#tab + 1);

    return ($max, $avg, $tab[$#tab]);
}


####### Maximum_WEEK_MONTH_YEAR
#  Oblicza maximum dla wyresow tygodniowego, miesiecznego i rocznego
#  @tab -- tablica danych do oblicznia
#######
sub maxWMY {
    my @tab = @_;
```

```
    my $max;
    my $q;

    if (defined($tab[0])) {
        $max = $tab[0]
    } else {
        return;
    }
    foreach my $i (@tab) {
         if ($max < $i) {
             $max = $i }
    }
    return $max;
}


####### Przelicz_na_bity
#  Przelicza dane z bajtow/s na bity/s (lub kb/s, Mb/s)
#  $odczyt  -- wartosc ruchu w B/s, do przeliczenia
#  $pred    -- predkosc podlaczenia w b/s lub "?"
#  $typ     -- typ danych
#  $miara_d -- miara danych
#######
sub przelicz {
    my $odczyt  = $_[0];
    my $pred    = $_[1];
    my $typ     = $_[2];
    my $miara_d = $_[3];
    my %druk;
    my $q;

    if ($typ == 5) {
        ########## CPU ##########
        $druk{'jednostka'} = $miara_d;
        $druk{'wartosc'} = $odczyt;
    } elsif ($typ == 4 && $miara_d eq "cells") {
        ########## CELLKI ##########
         $druk{'jednostka'} = "Cells/s";
         $druk{'wartosc'} = $odczyt;
        #if ($pred ne "?") {
         #    $druk {'procent'} = $odczyt / bit_cell($pred) ;
         #    $druk {'procent'} *= 100;
         #}
    } else {     # typ < 5 & miara = bits
        if ($typ == 4) {
             $odczyt *= 53;                         # teraz mamy tu juz BAJTy
         }
        if ($pred ne "?") {
            $pred /= 8;
            $druk {'procent'} = $odczyt / $pred ;  # jaki procent PASMA stanowi
ODCZYT
             $druk {'procent'} *= 100;
        }
        if ($odczyt > 256000) {
                                        # 2 Mb/s = 2048000 b/s = 256000 B/s
            $druk{'jednostka'} = "Mb/s";
            $druk{'wartosc'} = $odczyt / 128000 ;
                                        # B/s = 8/1024000 Mb/s = 1/128000 Mb/s
        } elsif ($odczyt > 1200) {
                                        # 9.6 kb/s = 9600 b/s = 1200 B/s
            $druk{'jednostka'} = "kb/s";
            $druk{'wartosc'} = $odczyt / 125 ;
                                        # B/s = 8/1000 kb/s = 1/125 kb/s
        } else {
            $druk{'jednostka'} = "b/s";
            $druk{'wartosc'} = $odczyt * 8 ;
                                        # B/s = 8 b/s
        }
```

```
    }
    return %druk;
}


####### Zamiana_bitow_na_cellki
##  Zamiana predkosci w bitach na cellki
##  $_[0] -- predkosc w bitach
########
sub bit_cell {
    my $pom;

    $pom = ($_[0] - ($_[0] % 53)) / 53;
    return $pom;
}


####### Generuje_statystyki
#  Umieszcza wyliczane statystyki z opisami w hashu
#  $hash    -- referencja do hasha tablic z danymi z rrdtool fetch
#  $typ     -- typ danych
#  $hashMAX -- referencja do hasha tablic MAXIMOW z danymi z rrdtool fetch
#              dla okresu YEAR
#######
sub stats {
    my $hash = $_[0];
    my $typ  = $_[1];
    my @wyn;
    my @pom;
    my %wynik;
    my $q;

    if ($typ == 5) {
        my @klucz = qw(max avg cur);
        @pom = max_avg_last(@{ $hash->{'count'} });
        if ($okres ne 'day') {
            my $hashMAX = $_[2];
            $pom[0] = maxWMY(@{ $hashMAX->{'count'} });
        }
        push ( @wyn, @pom );
        for ( $i=0; $i<3; $i++ ) {
            %{ $wynik{$klucz[$i]}} = przelicz($wyn[$i],$speed,$typ_zb,$miara);
        }
    } else {
        my @klucz = qw(maxin avgin curin maxout avgout curout);
        @pom = max_avg_last(@{ $hash->{'in'} });
        if ($okres ne 'day') {
            my $hashMAX = $_[2];
            $pom[0] = maxWMY(@{ $hashMAX->{'in'} });
        }
        push ( @wyn, @pom);
        @pom = max_avg_last(@{ $hash->{'out'} });
        if ($okres ne 'day') {
            my $hashMAX = $_[2];
            $pom[0] = maxWMY(@{ $hashMAX->{'out'} });
        }
        push ( @wyn, @pom);
        for ( $i=0; $i<6; $i++ ) {
            %{ $wynik{$klucz[$i]}} = przelicz($wyn[$i],$speed,$typ_zb,$miara);
        }
    }

    return %wynik;
}


####### Generuje_wydruk
#  Przygotowuje wydruk statystyk "pod wykres" oraz MAX do Gornej_Granicy_Wykresu
```

```
#  %hash -- hasz haszy z danymi do druku
#######
sub wydruk {
    my $hash    = $_[0];
    my $pred    = $_[1];
    my $typ     = $_[2];
    my $miara_d = $_[3];
    my $max;

    if ($typ == 5) {
        printf "%.0f;;\n<TABLE CELLPADDING=0 CELLSPACING=0><TR>
             <TD><SMALL><FONT COLOR=\"#00cc00\">Max:  </FONT></SMALL></TD>
            <TD ALIGN=right><SMALL>%5.2f %s    </SMALL></TD><TD
WIDTH=5></TD>\n
             <TD><SMALL><FONT COLOR=\"#00cc00\">Average:  </FONT></SMALL></TD>
             <TD ALIGN=right><SMALL>%5.2f %s    </SMALL></TD><TD
WIDTH=5></TD>\n
             <TD><SMALL><FONT COLOR=\"#00cc00\">Current:  </FONT></SMALL></TD>
             <TD ALIGN=right><SMALL>%5.2f %s   
</SMALL></TD></TR></TABLE>",
             $hash->{'max'}->{'wartosc'},
             $hash->{'max'}->{'wartosc'},$hash->{'max'}->{'jednostka'},
             $hash->{'avg'}->{'wartosc'},$hash->{'avg'}->{'jednostka'},
             $hash->{'cur'}->{'wartosc'},$hash->{'cur'}->{'jednostka'};
    } else {
        $max = znajdzMAX($hash);
        printf "%.0f;;\n<TABLE CELLPADDING=0 CELLSPACING=0><TR><TD><SMALL>Max
            <FONT COLOR=\"#00cc00\"> In:   </FONT></SMALL></TD>
            <TD ALIGN=right><SMALL>%5.2f %s ",
            $max,$hash->{'maxin'}->{'wartosc'},$hash->{'maxin'}->{'jednostka'};
        if ($pred ne "?" && $typ < 5 && $miara_d ne "cells") {
            printf "(%3.1f%%)  ",$hash->{'maxin'}->{'procent'};
        }
        printf "</SMALL></TD><TD WIDTH=5></TD>\n<TD><SMALL>Average
            <FONT COLOR=\"#00cc00\"> In:   </FONT></SMALL></TD>
             <TD ALIGN=right><SMALL>%5.2f %s ",
             $hash->{'avgin'}->{'wartosc'},$hash->{'avgin'}->{'jednostka'};
        if ($pred ne "?" && $typ < 5 && $miara_d ne "cells") {
            printf "(%3.1f%%)  ",$hash->{'avgin'}->{'procent'};
        }
        printf "</SMALL></TD><TD WIDTH=5></TD>\n<TD><SMALL>Current
            <FONT COLOR=\"#00cc00\"> In:   </FONT></SMALL></TD>
             <TD ALIGN=right><SMALL>%5.2f %s ",
             $hash->{'curin'}->{'wartosc'},$hash->{'curin'}->{'jednostka'};
        if ($pred ne "?" && $typ < 5 && $miara_d ne "cells") {
            printf "(%3.1f%%)  ",$hash->{'curin'}->{'procent'};
        }
        printf "</SMALL></TD></TR><TR><TD ALIGN=right><SMALL>Max
            <FONT COLOR=\"#0000ff\"> Out: </FONT></SMALL></TD>
             <TD ALIGN=right><SMALL>%5.2f %s ",
             $hash->{'maxout'}->{'wartosc'},$hash->{'maxout'}->{'jednostka'};
        if ($pred ne "?" && $typ < 5 && $miara_d ne "cells") {
            printf "(%3.1f%%)  ",$hash->{'maxout'}->{'procent'};
        }
        printf "</SMALL></TD><TD WIDTH=5></TD>\n<TD ALIGN=right><SMALL>Average
            <FONT COLOR=\"#0000ff\"> Out: </FONT></SMALL></TD>
             <TD ALIGN=right><SMALL>%5.2f %s ",
             $hash->{'avgout'}->{'wartosc'},$hash->{'avgout'}->{'jednostka'};
        if ($pred ne "?" && $typ < 5 && $miara_d ne "cells") {
            printf "(%3.1f%%)  ",$hash->{'avgout'}->{'procent'};
        }
        printf "</SMALL></TD><TD WIDTH=5></TD>\n<TD ALIGN=right><SMALL>Current
            <FONT COLOR=\"#0000ff\"> Out: </FONT></SMALL></TD>
             <TD ALIGN=right><SMALL>%5.2f %s ",
             $hash->{'curout'}->{'wartosc'},$hash->{'curout'}->{'jednostka'};
        if ($pred ne "?" && $typ < 5 && $miara_d ne "cells") {
            printf "(%3.1f%%)  ",$hash->{'curout'}->{'procent'};
        }
```

```
        print "</SMALL></TD></TR></TABLE>\n";
    }
}


####### Znajdz_maximum
#  Znajduje maximum maaximow 'in' i 'out'
#  %hash -- hasz haszy z danymi do druku
#######
sub znajdzMAX {
    my $hash = $_[0];
    my $max;
    my $pom;

    if ($hash->{'maxin'}->{'jednostka'} eq "Mb/s") {
        $pom = $hash->{'maxin'}->{'wartosc'} * 1024000;
    } elsif ($hash->{'maxin'}->{'jednostka'} eq "kb/s") {
        $pom = $hash->{'maxin'}->{'wartosc'} * 1000;
    } else {
        $pom = $hash->{'maxin'}->{'wartosc'};
    }
    if ($hash->{'maxout'}->{'jednostka'} eq "Mb/s") {
        $max = $hash->{'maxout'}->{'wartosc'} * 1024000;
    } elsif ($hash->{'maxout'}->{'jednostka'} eq "kb/s") {
        $max = $hash->{'maxout'}->{'wartosc'} * 1000;
    } else {
        $max = $hash->{'maxout'}->{'wartosc'};
    }
    if ($max < $pom) {
        $max = $pom;
    }
    return $max;
}
```

**Skrypt *view.pl***

```
#!/usr/bin/perl -w
################################################################
#                              VIEW.PL                         #
#     utworzenie tymczasowego WYKRESU za podany okres          #
#   $1 -- krotka_nazwa_klienta bez sciezki/zrozszerzenia   #
#   $2 -- predkosc klienta w b/s                             #
#   $3 -- maksimum ruchu w okresie_rozliczeniowym            #
#   $4 -- okres_rozliczeniowy                                #
#   $5 -- "aktualny" czas bazy - ostatnie uaktualnienie RRD#
#   $6 -- katalog z RRD                                      #
#   $7 -- typ danych                                         #
#   $8 -- jednostka danych                                   #
#   $9 -- zwrot danych                                       #
#                                                            #
################################################################


use strict;
use CGI qw(param);
use POSIX qw(strftime);
use Time::Local;
use Date::Manip qw(ParseDate UnixDate);

my $cgi = new CGI;

my $nazwa  = $cgi->param("nazwa");                # nazwa klienta
my $speed  = $cgi->param("predkosc");             # predkosc klienta
my $max    = $cgi->param("max");          # rzeczywiste maximum w okresie
my $okres  = $cgi->param("okres");                # okres wykresu
my $czas   = $cgi->param("czas");          # aktualny czas
my $rrdDIR = $cgi->param("katalog");              # katalog z RRD
my $typ    = $cgi->param("typ_zb");               # typZbierania bajt/cell/inne
my $miara  = $cgi->param("miara");                # jednostki: bits/cells/percent
my $zwrot  = $cgi->param("zwrot");                # zwrot danych: 0-ok, 1-odwroc

my $dir;
my $konf;
my $opis;
my $linie;
my $siatka;
my $wykres;
my $zmienne;


################################################
#####    GLOWNY_poczatek

{
    print "Content-type: image/png\n\n";
    my $z1 = $czas - ($czas % 300);                       # koniec okresu
    my $z2;                                               # poczatek okresu
    my $druk;

    $dir  = plikRRD($nazwa,$rrdDIR);
    $konf = opcjeSTD();          # w h a l g c
    $opis = opcjeETYK($miara);

    if ($miara eq "cells" && $typ != 4) {
        print STDERR "VIEW.PL\tGlowny: niezgodnosc miara= $miara i $typ!\n";
        die;
    }
    if ($okres eq 'day') {
        $z2 = $z1 - 150000;
        $linie   = opcjeLINIE($okres,$speed,$max,$miara);
        $siatka  = opcjeSIATKA($okres,$max,$speed);
```

```
        $zmienne = defZMIENNE($okres,$dir,$zwrot,$typ,$miara);
        $wykres  = defWYKRES($okres,$typ);

                # $KONF    -- opcje standardowe: W H A L G C
                # $OPIS    -- opcje etykiety
                # $LINIE   -- opcje dot. linii specjanych (krance okresow, maximum)
                # $SIATKA  -- opcje dot. siatki
                # $WYKRES  -- definiowanie wykresu
                # $ZMIENNE -- definiowanie zmiennych
        print `rrdtool graph - -s $z2 -e $z1 $konf $linie $siatka $opis $zmienne
$wykres`

    } elsif ($okres eq 'week') {
        $z2 = $z1 - 900000;
        $linie   = opcjeLINIE($okres,$speed,$max,$miara);
        $siatka  = opcjeSIATKA($okres,$max,$speed);
        $zmienne = defZMIENNE($okres,$dir,$zwrot,$typ,$miara);
        $wykres  = defWYKRES($okres,$typ);

        print `rrdtool graph - -s $z2 -e $z1 $konf $linie $siatka $opis $zmienne
$wykres`;

    } elsif ($okres eq 'month') {
        $z2 = $z1 - 3600000;
        $linie   = opcjeLINIE($okres,$speed,$max,$miara);
        $siatka  = opcjeSIATKA($okres,$max,$speed);
        $zmienne = defZMIENNE($okres,$dir,$zwrot,$typ,$miara);
        $wykres  = defWYKRES($okres,$typ);

        print `rrdtool graph - -s $z2 -e $z1 $konf $linie $siatka $opis $zmienne
$wykres`;

    } else {
        $z2 = $z1 - 43200000;
        $linie   = opcjeLINIE($okres,$speed,$max,$miara);
        $siatka  = opcjeSIATKA($okres,$max,$speed);
        $zmienne = defZMIENNE($okres,$dir,$zwrot,$typ,$miara);
        $wykres  = defWYKRES($okres,$typ);

        print `rrdtool graph - -s $z2 -e $z1 $konf $linie $siatka $opis $zmienne
$wykres`;
    }

my @html;
    push(@html, $druk);
        $" = "";
            #print "Content-type: text/html\n\n";
               # print "@html";
}

#####   GLOWNY_koniec
################################################


####### Opcje_standardowe
#  Ustala wspolne dla wszystkich okresow opcje
#######
sub opcjeSTD {
    my $opcje = " ";

    $opcje .= "-w 500 -h 150 ";                             # wielkosc obrazka
    $opcje .= "-a PNG ";                            # format obrazka
    $opcje .= "-l 0 ";                                      # dolny kraniec obrazka
    $opcje .= "-g ";                                        # bez legendy
    $opcje .= "-c GRID#333333 -c MGRID#333333 ";   # obramowanie wykresu
    $opcje .= "-c SHADEB#555555 -c ARROW#000000 ";          # obramowanie wykresu

    return $opcje;
```

```
}


####### Opcje_Linii
#  Okresla opcje dotyczace linii poziomych i pionowych
#  $_[0] -- okres wykresu
#  $_[1] -- predkosc
#  $_[2] -- maximum
#  $_[3] -- miara
#######
sub opcjeLINIE {
    my @pom;
    my $opcje;

    @pom = wertykalne($_[0]);
    $opcje = "VRULE:$pom[0]#FF2222 VRULE:$pom[1]#FF2222 ";
    if ($_[1] ne "?" && $_[3] ne "cells" && $_[2] > $_[1]) {
    #    $_[1] *=1.002;
        $opcje .= "HRULE:$_[1]#FF2222 ";
    }
    return $opcje;
}


####### Opcje_Siatki
#  Okresla opcje "-x" i "-y" dotyczaca siatki
#  $_[0] -- okres wykresu
#  $_[1] -- maximum
#  $pred -- predkosc
#######
sub opcjeSIATKA {
    my $max  = $_[1];
    my $pred = $_[2];
    my $opcje;
    my $op1;
    my $op2;
    my $pom;

#print STDERR "max:   $max\n\n";
#print STDERR "pred:  $pred\n\n";
    if ($_[0] eq 'day') {
       $opcje = "-x HOUR:1:DAY:1:HOUR:2:0:%k ";
    } elsif ($_[0] eq 'week') {
       $opcje = "-x DAY:1:WEEK:1:DAY:1:86400:%a ";
    } elsif ($_[0] eq 'month') {
       $opcje = "-x WEEK:1:MONTH:1:WEEK:1:604800:'Week '%V ";
    } else {
       $opcje = "-x MONTH:1:YEAR:1:MONTH:1:2628000:%b ";
    }
    if ($pred ne "?") {
        if ( ($max/$pred < 0.95 && $max != 0) || $max > $pred) {
                #if ( $max/$pred < 0.95 && $_[0] ne 'year' && $max != 0)
            $pom = $max;
            $pom *= 0.25;
             $op1 = " -y $pom:1 ";
        } elsif ($max == 0) {
             $pom = 1;
             $op1 = " -y $pom:1 ";
         } else {
            $pom = $pred;
             $pom *= 0.25;
             $op1 = " -y $pom:1 ";
        }
        if ($max < $pred && $max/$pred > 0.95 ) {
                #if ($max > $pred || ( $max/$pred < 0.95 && $_[0] ne 'year'))
            $pred *=1.002;
             $op2 = "-u $pred -r ";
        } elsif ($max == 0) {
```

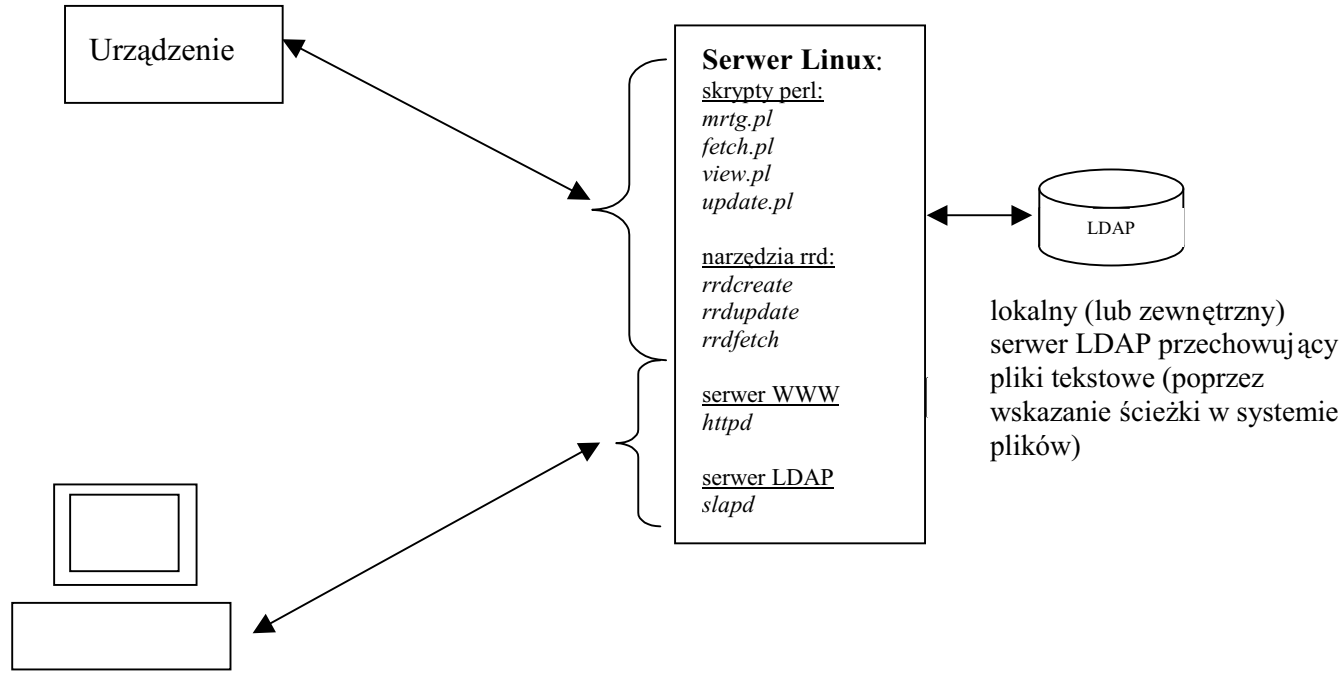
```
            $max = 4;
            $op2 = "-u $max -r ";
        #} elsif ($max > $pred) {
        #    $pred *=1.002;
        #    $op2 = "-u $pred -r ";
        } else {
            $max *=1.002;
            $op2 = "-u $max -r ";
        }
    } elsif ($max != 0) {
        $pom = $max;
        $pom *= 0.25;
        $op1 = " -y $pom:1 ";
        $max *=1.1;
        $op2 = "-u $max -r ";
    } else {
        $pom = 1;
        $op1 = " -y $pom:1 ";
        $pom = 4;
        $op2 = "-u $pom -r ";
    }

    $opcje .= $op1;
    $opcje .= $op2;
    #print STDERR "\n\n$opcje\n\n";
    return $opcje;
}


####### Opcje_Etykiety
#  Okresla opis danych na wykresie
#  $_[0] -- miara
#######
sub opcjeETYK {
    my $miara_d = $_[0];
    my $opcje   = " -v \"";

    $miara_d =~ s/(\S)/\U$1/;
    $opcje .= $miara_d;
    if ($_[0] eq "bits" || $_[0] eq "cells") {
        $opcje .= " per Second\" ";
    } else {
        $opcje .= " at time\" ";
    }
    return $opcje;
}


####### Definiuje_Zmienne
#  Definiuje zmienne przedstawiane na wykresie
#  $term -- okres
#  $kata -- sciezka do bazy RRD
#  $kier -- zwrot danych
#  $typD -- typ danych
#  $znak -- miara danych
#######
sub defZMIENNE {
    my $term = $_[0];
    my $link = $_[1];
    my $kier = $_[2];
    my $typD = $_[3];
    my $znak = $_[4];
    my $opcje;

    if ($typD < 5) {
        if ($kier == 0) {
            $opcje  = "DEF:inoct=$link:input:AVERAGE ";
            $opcje .= "DEF:outoct=$link:output:AVERAGE ";
```

```
            if ($term ne "day") {
                $opcje .= "DEF:imax=$link:input:MAX ";
                $opcje .= "DEF:omax=$link:output:MAX ";
            }
        } else {
            $opcje  = "DEF:inoct=$link:output:AVERAGE ";
            $opcje .= "DEF:outoct=$link:input:AVERAGE ";
            if ($term ne "day") {
                $opcje .= "DEF:imax=$link:output:MAX ";
                $opcje .= "DEF:omax=$link:input:MAX ";
            }
        }
        if ($typD < 4) {
            $opcje .= "'CDEF:inBite=inoct,8,*' ";
            $opcje .= "'CDEF:outBite=outoct,8,*' ";
            if ($term ne "day") {
                $opcje .= "'CDEF:inMax=imax,8,*' ";
                $opcje .= "'CDEF:outMax=omax,8,*' ";
            }
        } elsif ($znak eq "bits") {
            $opcje .= "'CDEF:inBite=inoct,424,*' ";
            $opcje .= "'CDEF:outBite=outoct,424,*' ";
            if ($term ne "day") {
                $opcje .= "'CDEF:inMax=imax,424,*' ";
                $opcje .= "'CDEF:outMax=omax,424,*' ";
            }
        } else {
            $opcje .= "'CDEF:inBite=inoct,1,*' ";
            $opcje .= "'CDEF:outBite=outoct,1,*' ";
            if ($term ne "day") {
                $opcje .= "'CDEF:inMax=imax,1,*' ";
                $opcje .= "'CDEF:outMax=omax,1,*' ";
            }
        }
    } elsif ($typD == 5) {
       $opcje  = "DEF:countAve=$link:count:AVERAGE ";
       if ($term ne "day") {
           $opcje .= "DEF:countMax=$link:count:MAX ";
       }
    } else {
       print STDERR "VIEW.PL\tdefZMIENNE: typ_danych > 5 !!!\n";
       die;
    }
    return $opcje;
}


####### Definiuje_Wykres
#  Definiuje elementy wykresu
#  $term -- okres
#  $typD -- typ danych
#######
sub defWYKRES {
    my $term = $_[0];
    my $typD = $_[1];
    my $opcje;

    if ($typD < 5) {
        if ($term ne "day") {
            $opcje  = "AREA:inMax#006600 AREA:inBite#00CC00 ";
            $opcje .= "LINE1:outMax#FF00FF LINE1:outBite#0000FF ";
        } else {
            $opcje = "AREA:inBite#00CC00 LINE1:outBite#0000FF ";
        }
    } elsif ($typD == 5) {
        if ($term ne "day") {
            $opcje = "AREA:countMax#006600 AREA:countAve#00CC00 ";
        } else {
```

```
            $opcje = "AREA:countAve#00CC00 ";
        }
    } else {
        print STDERR "VIEW.PL\tdefWYKRES: typ_danych > 5 !!!\n";
        die;
    }
    return $opcje;
}


####### Plik_rrd
#  Okresla pelna sciezke do pliku RRD
#  $kli -- klient
#  $kat -- katalog z plikami RRD
#######
sub plikRRD {
    my $kli = $_[0];
    my $kat = $_[1];

    $kat .= $kli;
    $kat .= ".rrd";
    return $kat;
}


####### Wertykalne_linialy
#  Okresla pionowe linialy
#  $_[0] -- okres wykresu
#######
sub wertykalne {
    my ($h1,$h2);
    my $chw;
    my ($day,$month,$year);

    ($day, $month, $year) = (localtime)[3,4,5];
    if ($_[0] eq 'day') {
        $h1 = timelocal (0,0,0, $day, $month, $year);
        $h2 = $h1 - 86400;                                  # druga linia
DZIEN_WCZESNIEJ
    } elsif ($_[0] eq 'week') {
        $chw = (localtime)[6];                              # aktualny dzien
tygodnia
        $chw -= 1;
        $h1 = timelocal (0,0,0, $day, $month, $year);
        $h1 = $h1 - $chw * 86400;
        $h2 = $h1 - 604800;                                 # druga linia
TYDZIEN_WCZESNIEJ
    } elsif ($_[0] eq 'month') {
        $h1 = timelocal (0,0,0,1, $month, $year);
        if ($month == 0) {
            $month = 12;
            $year = $year - 1;
        }
        $h2 = timelocal (0,0,0,1, $month - 1, $year);
    } else {
        $h1 = timelocal (0,0,0,1,0, $year);
        $h2 = timelocal (0,0,0,1,0, $year - 1);
    }
    return ($h1,$h2);
}


####### Horyzontalna_siatka
#  Okresla siatke pozioma
#  $v -- $speed
#  $m -- $max
#######
sub horizontal {
```

```
    my $v = $_[0];
    my $m = $_[1];
    my ($a,$b,$c);

    if ($m > ($v * 0.9) ) {
         $a = $v / 4;
         $c = $v * 1.015625;
        $b = $c - ($c % 10000) + 10000;
    } else {
        my $pom = $m - ($m % 10000) + 10000;
         $a = $pom / 4;
         $c = $pom * 1.015625;
         $b = $c - ($c % 10000) + 10000;
    }
    return ($a,$b);
}
```

## 6 Zastosowanie serwera LDAP w procesie zbierania i udostępniania danych o stanie urządzeń



# 7 Zalety wdrożenia serwera LDAP do systemu monitorowania styków sieci POL -34

Do głównych zalety zastowania serwera LDAP jako bazy plików tekstowych zawierających uaktualnienia można zaliczyć:

- Znacznie szybszy i bardziej efektywny mechanizm dostępu i obsługi plików (odczyt, przeglądanie, wyszukiwanie)
- Brak konieczności generowania stron HTML do prezentacji utworzonych plików
- Możliwość dostępu do danych za pomocą protokół LDAP przy wykorzystaniu przeglądarki lub ogólnodostępnych programów typu LDAP agent
- Łatwość dalszej replikacji danych w oparicu o mechanizmy LDAP, np. dla potrzeb archiwizacji
- Możliwość zastosowania dostępnych dla LDAPa mechanizmów bezpieczeństwa takich jak:
    - Uwierzytelnienie użytkownika w oparciu o hasło
    - Uwierzytelnienie strony agenta przy dostępie do serwera
    - Autoryzacja w oparciu o mechanizm Access Control List
    - Zastosowanie SASL (Simple Authentication and Security Layer)

# 8 Sposób korzystania systemu z bazy katalogowej LDAP

Z uwagi na konieczność zapewnienia efektywnego dostępu do plików tekstowych niezbędnych dla generacji statystyk, baza LDAP przechowuje informacje o ich położeniu (ścieżki). W pierwszym - testowym etapie, zdecydowaliśmy się na użycie LDAPa jedynie jako bazy indeksującej.

## 8.1 Struktura bazy LDAP

Poniżej testowa schema realizująca zadanie przechowywania informacji o plikach.

***# schema dla przechowywania informacji o położeniu plików***
***#***
***attributetype (1.3.6.1.4.1.15373.2.0.0 NAME 'datafile'***
***DESC 'path to MRTG RRD datafile'***
***EQUALITY IA5Match***
***SYNTAX 1.3.6.1.4.1.1466.115.121.1.26 )***

## 8.2 Informacja o zastosowanych klasach i atrybutach

Zastosowane klasy obiektów i atrybuty

*attributetype (1.3.6.1.4.1.15373.2.0.0 NAME 'datafile'*

Poszczególne komponenty identyfikatora OID:

| | |
|---|---|
| ISO assigned | 1 |
| Organization acknowledged by ISO | 3 |
| US Department of Defence | 6 |
| Internet | 1 |
| Private | 4 |
| IANA registered private enterprises | 1 |
| NASK | 15373 |

*IA5 String*

*( 1.3.6.1.4.1.1466.115.121.1.26 DESC 'IA5 String' )*

The encoding of a value in this syntax is the string value itself.