Ponmon

Модуль моніторингу PON обладнання

- Модуль написаний для особистого використання і з моєї доброти буде доступний до покупки!
- Підтримується тільки те обладнання, яке було в наявності або доступне до тестування!
- Функціонал та підтримка нового обладнання нарощуватиметься, але не швидко ;)

Функціонал (Хотелі)

“+” - готове

“-” - Не готово

- + Пошук ОНУ за ОЛТ та занесення її до БД білінгу
- + Моніторинг стану та згасання ОНУ
- + Графіки загасання ОНУ
- + Отримання FDB таблиці ОЛТ та зіставлення з ОНУ
- + Відображення у картці абонента за якою ОНУ він сидить (динамічно). Це не прив'язка і не факт, що за ОНУ немає свіча))
- + Пошук ОНУ за серійним номером або за МАКом
- + Фільтр ОНУ по дереву
- - **Прив'язка ОНУ до абонента \ точки топології**
- - **Пошук по ТКД**
- - **Фільтр загасання РХ**
- - *можливість винести на окрему базу даних*

- - Моніторинг стану ОЛТ
- - Інформація про ПОН СФП
- Інші вендори

буде доповнюватись! поки все що згадав)

Backend

Модуль складається з трьох основних елементів:

Модуль ядра

```
./kermel/ponmon.pm
```

елемент займається створенням потоків моніторингу для кожної ОЛТ, збиранням результатів повернутих від ОЛТ, зіставленням та занесенням змін до БД, зіставленням авторизацій абонентів з таблицею FDB

Модуль вендора

```
./nod/Pon/Bdcom.pm
```

Збірник оідів та функцій по роботі з даним вендором. При отриманні завдання від модуля ядра, залежно від **кроку**, виконує функції отримання та конвертування даних у єдину структуру та повертає хеш із готовими даними модулю ядра.

Якщо ім'я функції починається з “_” нижнього підкреслення, то ця функція локальна і не повинна викликатися з-за, а використовується виключно самим модулем вендора.

Якщо ім'я починається з літери (поки таких немає, зачепив на майбутнє), то цю функцію можна викликати з-за, наприклад з

Інтернету для ребута ОНУ

Для отримання даних з SNMP використовує модуль ./nod/snmptool.pm

Модуль SNMP

```
./nod/snmptool.pm
```

Модуль роботи з протоколу SNMP На вхід отримує ХЕШ параметрів ОЛТ, ХЕШ оідів, ім'я команди (walk/get/set) На виході віддає отриманий результат

snmp.pl

У поставці з головним модулем буде файл snmp.pl (аналог системного snmpwalk), за допомогою якого можна побачити структуру з якою працює головний модуль, щоб покращити або додати модулі вендорів

Frontend

stat.pl?a=ponmon - центральна сторінка модуля ПЗН. Відображає список ОНУ без сортування за ОЛТ, пошук ОНУ за серійним номером (gpon) або маком (epon). Також звідси вхід на сторінку бінда ОНУ+ОЛТ де відображено інформацію про ОНУ, графік та функції з ОНУ.

stat.pl?a=op&op=list&act=olt_list - сторінка управління ОЛТ

На сторінці картки абонента під основним блоком відображається блок PON якщо абонент за ОНУ

Модулі вендорів PON моніторингу

Оскільки модуль моніторингу сам по собі не вміє нічого моніторити, йому потрібні модулі вендорів! Така структура була придумана для універсальності, і для кожного вендора можуть бути розроблені свої модулі, бо кожен має свої OIДи і алгоритми.

Тестувати та шукати OIДи дуже затратно за часом, тому приймаю інформаційну допомогу!!!

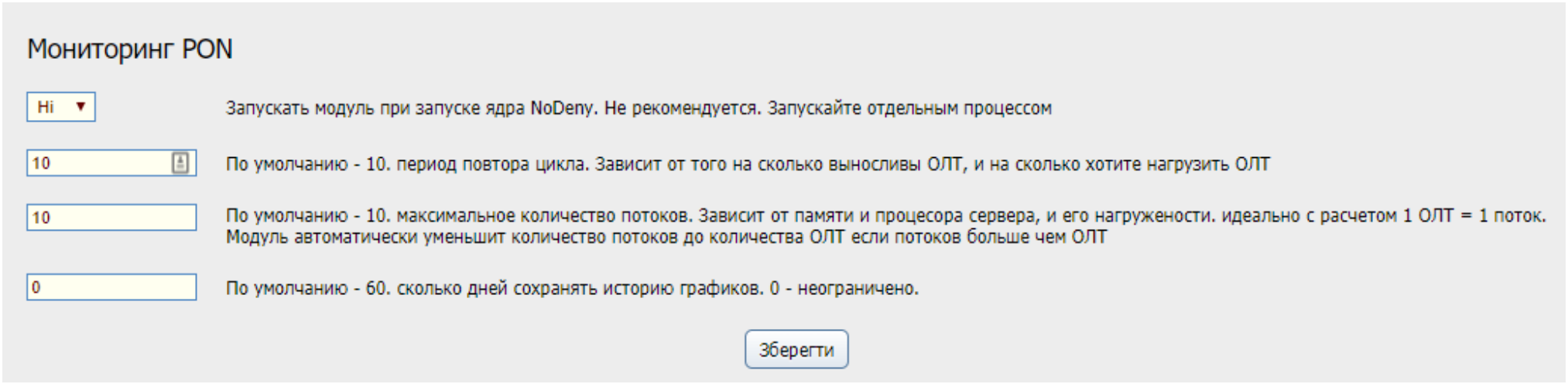
Також допоможу розробити модулі для інших вендорів, але потрібно десь тестувати  !

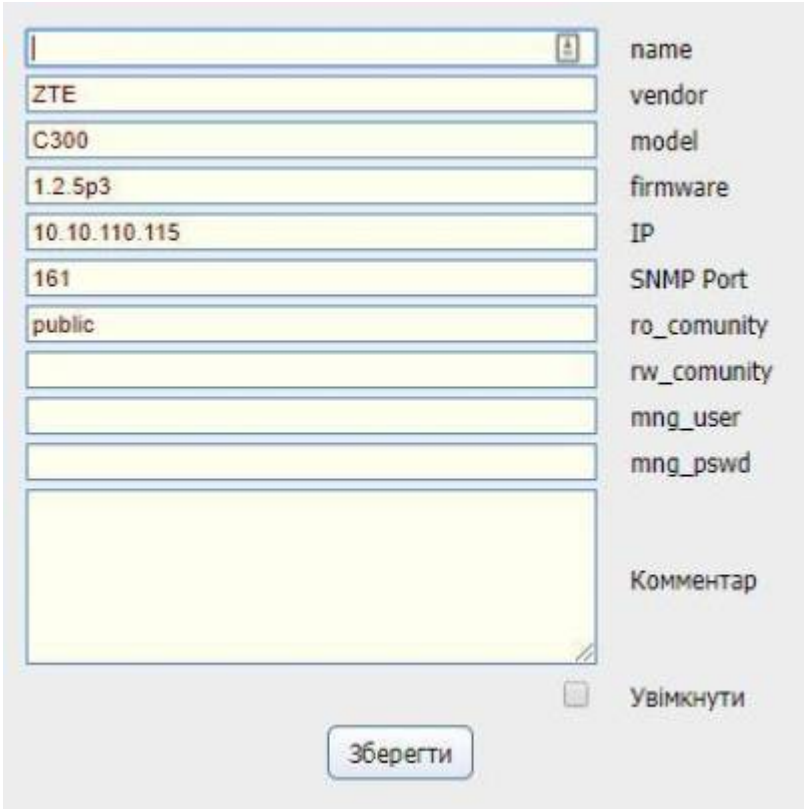
Інструкція

Для роботи модуля необхідно встановити пакет для SNMP та модулі PERLa

```
pkg install net-snmp p5-Net-SNMP p5-Net-SNMP-Util p5-Net-Telnet-Cisco p5-Net-Ping p5-Parallel-ForkManager
```

У налаштуваннях білінгу, Ядро → Моніторинг PON змінити під себе та зберегти





У налаштуваннях білінгу, пункт “OLT Сервера” додати свої ОЛТ

пробний запуск командою

```
/usr/bin/perl /usr/local/nodeny/nokernel.pl -m=ponmon -v
```

ставимо в автозавантаження:

```
echo '/usr/bin/perl /usr/local/nodeny/nokernel.pl -m=ponmon -d &' >> /etc/rc.local
```

Для читання FDB таблиць по телнету необхідно додати до документів шаблон налаштувань і вибрати його в налаштуваннях конкретної OLT

Шаблон налаштувань повинен мати теги system та pon_tmpl

Атрибути для різних вендорів OLT відрізняються, тому, щоб дізнатися які атрибути доступні, можна скористатися командою

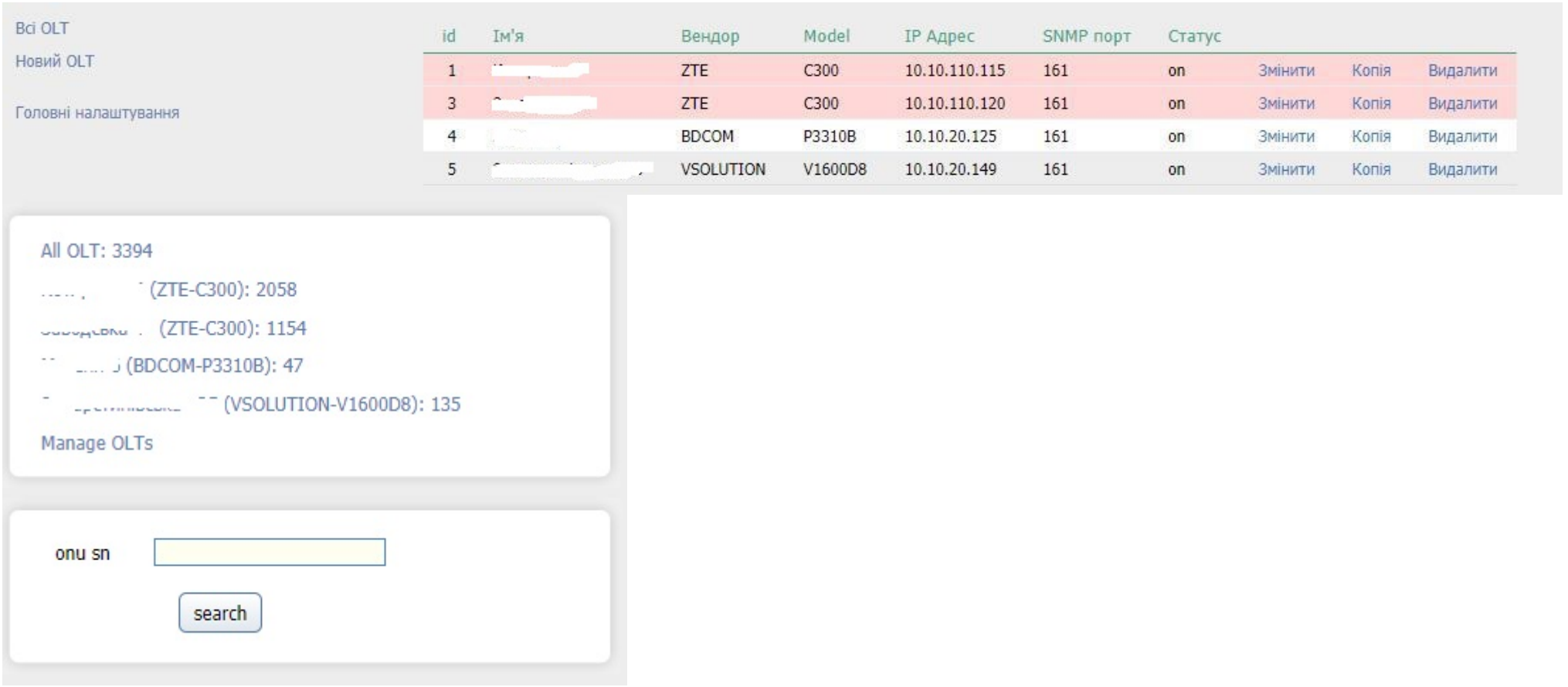
```
cat /usr/local/nodeny/nod/Pon/_Zte.pm | grep "{cfg}"
```

і брати до уваги все, що в наступних фігурних дужках. наприклад з рядка “*if ($olt→{cfg}{fdb_force_telnet}) {*” беремо **fdb_force_telnet**

Надалі відмовлятимуся від ідеї з використанням шаблонів і перенесу еті налаштування в конкретну OLT!!!

Також буде можливість вносити свої кастомні атрибути та використовувати їх у своїх додаткових патчах.

Скріншоти


Всі OLT
Новий OLT
Головні налаштування
id Ім'я Вендор Model IP Адрес SNMP порт Статус
1 ZTE C300 10.10.110.115 161 on Змінити Копія Видалити
3 ZTE C300 10.10.110.120 161 on Змінити Копія Видалити
4 BDCOM P3310B 10.10.20.125 161 on Змінити Копія Видалити
5 VSOLUTION V1600D8 10.10.20.149 161 on Змінити Копія Видалити
All OLT: 3394
(ZTE-C300): 2058
(ZTE-C300): 1154
(BDCOM-P3310B): 47
(VSOLUTION-V1600D8): 135
Manage OLTs
onu sn
search

INFO	:	80:14:A8:4	-17.10	1.59	OnLine(1)	0:Wire down
INFO	:	80:14:A8:4	-22.01	1.72	OnLine(1)	1:Power off
INFO	:	80:14:A8:4	-27.45	2.01	OnLine(1)	1:Power off
INFO	:	80:14:A8:4		1.59	OnLine(1)	0:Wire down
INFO	:	80:14:A8:4			OffLine(0)	0:Wire down
INFO	:	80:14:A8:4		1.74	OnLine(1)	1:Power off
INFO	:	80:14:A8:4		1.35	OnLine(1)	1:Power off
INFO	:	80:14:A8:4		2.09	OnLine(1)	1:Power off
INFO	:	80:14:A8:4	-17.35	1.56	OnLine(1)	1:Power off
INFO	:	80:14:A8:4	-22.84	1.97	OffLine(0)	0:Wire down
INFO	:	80:14:A8:4	-17.26	1.86	OffLine(0)	1:Power off
INFO	:	80:14:A8:4		1.79	OnLine(1)	1:Power off
INFO	:	80:14:A8:4	-19.75	1.44	OnLine(1)	1:Power off
INFO	:	80:14:A8:5	-23.87	2.08	OnLine(1)	1:Power off
INFO	:	80:14:A8:5		1.99	OnLine(1)	1:Power off
INFO	:	80:14:A8:5		1.99	OnLine(1)	1:Power off
INFO	:	80:14:A8:5		1.85	OnLine(1)	0:Wire down

1 2 3

Авторизація	Ввімнута	Телефон дом.	...
Знижка, %	0	Точка топологии	...
Межа відключення	0	Технологія підключення	PON
		Аліас	0
		Посвідчення АТО/інваліда	...

PON

	MAC/IP	OLT	ONU	dBm	Status	Updated
INFO	D4:6[redacted] 10.10.155.225	[redacted]40	ZTEGC[redacted] gpon-onu_1/2/1:100	RX: -17.47 TX: 2.62	Online(3)	16.01.2021 19:22

Графіки

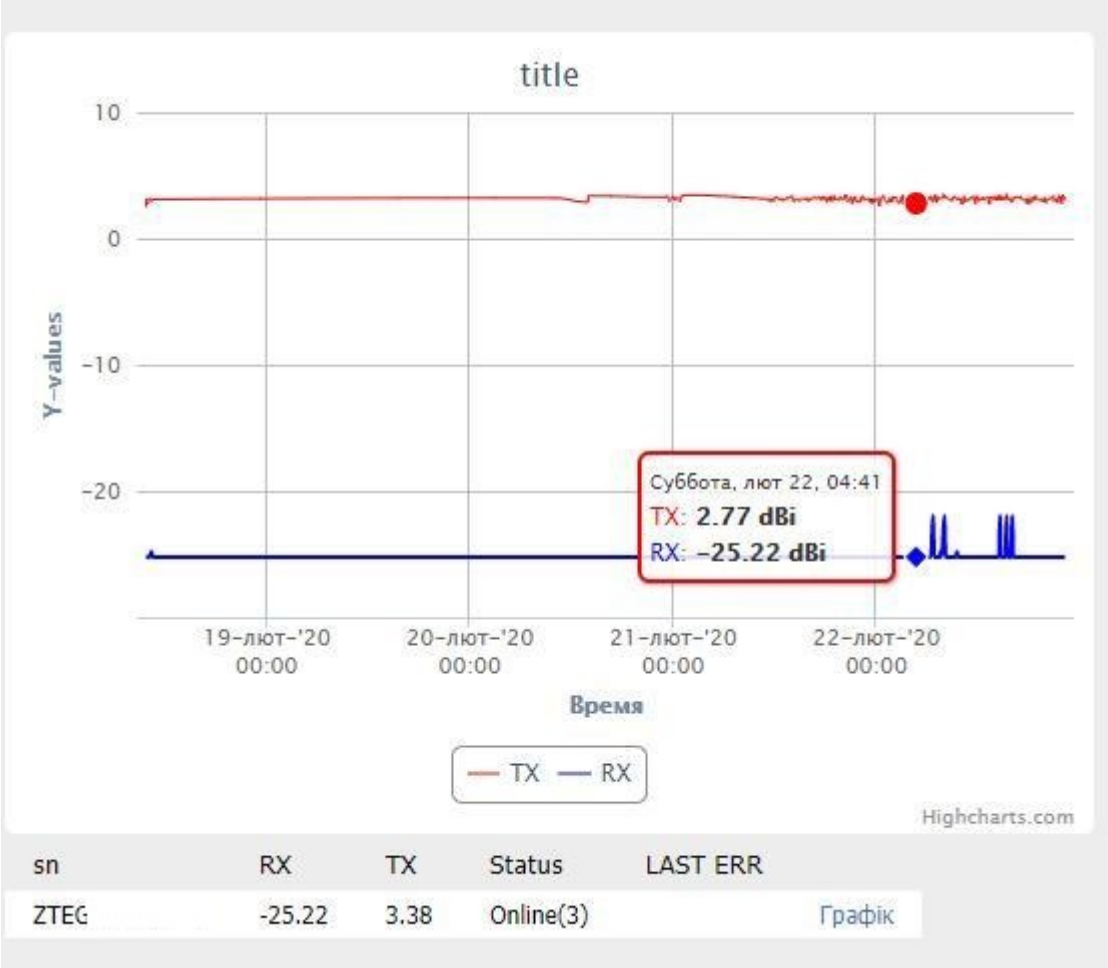
січень 2021: 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1
грудень 2020: 31 30 29 28 27 26 25 24 23 22 21 20 19 18 17 16 15 14 13 12 11 10 9 8 7 6 5 4 3 2 1

BIND info

sn	RX	TX	Status	LAST CHANGE	
ZTE[redacted]	-17.47	2.62	Online(3)	16.01.2021 19:22	Дополнительно

FDB CACHE

Клієнт	mac	vlan	ip	LAST CHANGE
[redacted]	D4:[redacted]	2001	10.10.155.225	16.01.2021 19:21



Передісторія

Якось мені на роботі було поставлено завдання знайти спосіб керування та моніторингу PON обладнанням. Спочатку було вирішено використовувати сторонній сервіс http://www.easypon.in/, тоді вони тільки починали розвиватися і послуга була повністю безкоштовна на тестування на нашому обладнанні. Але раптово вони виставили захмарний рахунок, не пам'ятаю точно, але дуже дохрена багато (близько 1.5 $ на місяць за ОНУ). Ми знали, що сервіс стане платним, але що б так дорого, навіть не підозрювали! Потім пішли ідеї навісити надбудову

UserSide, але ціна також не підійшла:)! Потім директор запропонував змінити білінг на той, у якого даний функціонал в наявності, але йому бік, скільки вкладено сил і часу на свої патчі для NoDeny Plus або скільки проблем принесе переїзд на новий білінг, а так само часу та грошей!

Саме тоді і народилася ідея написати свій модуль сабжа, в якому буде те, що нам потрібно і так як нам це потрібно! Вирішено! Починаємо! Як поважаючий себе програміст, я природно починаю гуглити)! Прошерстивши майже весь github, знаходжу потрібну мені інформацію (до речі 3 проекти з України!) і починаю писати модуль пона! На той момент ми мали близько семи ОЛТ від BDcom і V-Solution(Spectra) за технологією (G)EPON. З ними, в основному, все було просто, але я не забував про спільноту форумчан білінгу, і тому вирішив робити універсальну платформу для будь-якого вендора і технології, що потягнуло за собою вічні переробки в структурі модуля і БД до нього, і нарешті начебто як все склалося, фірма купує пон шасі ZTE C300 (шасі на 14 слотів з підтримкою технологій (G)EPON, GPON, xPON. залежить від встановлених у неї слотів), зі слотами під GPON. Крім того, вирішено перейти повністю з EPON на GPON і відмовитися від BDcom і V-Solution. І тут я розумію, що вся виконана робота на 90% стала марною! Я в смутку, але починаю розумітися на ZTE, а у них міби закриті! Як вдалося знайти частину мібів, деякі відсутні сам дописав.

Пішла робота наново!

А в порівнянні з BDcom та V-Solution, ZTE – мега-комбайн, який ну дуже класно продуманий! Я швиденько накотив модуль роботи з ним, але вибірка з БД стала дуже довгою, до того ж, треба все звести до єдиної структури! І тут знову затік! Начальство каже відкласти пон убік т.к. назбиралися інші завдання, що не терплять зволікання, типу заміна всіх мікротиків на Juniper MX80, використання телефонії, ЦРМ та інші завдання не до розголошення.

І ось настав час, я набрався мужності випросити місяць на продовження роботи з поном!

Подальшу історію не буду поки що описувати!

Підсумую:

- Загалом модуль переписувався майже з нуля 6 разів!
- На все витрачено понад 800 годин робочого та особистого часу!

Розробникам

Спілкування основного модуля з модулями вендорів відбувається за допомогою одного виклику головної функції вендора **main($olt, $step)**, де в змінній $olt передаються всі налаштування конкретної ОЛТ + вміст шаблону керування за наявності його, а змінної $step - просто номер кола сканування з моменту запуску головного модуля пона.

Змінна $step використовується для пропусків сканування не пріоритетних параметрів, якщо це критично даного вендора.

Функція **main** у свою чергу запускає послідовність сканування ОЛТ та повертає хеш результатів сканування

```
sub main {
    my $self = shift; # приймає ім'я модуля вендора -
    my $olt = shift; # приймає хеш налаштувань ОЛТ
    my $step = shift; # приймає номер кола сканування

    $olt->{snmp_v} ||= 2; # встановлює або керує версію протоколу  snmp
    $olt->{snmp_t} ||= 10; # встановлює або править таймаут відповідей  snmp
    $olt->{snmp_r} ||= 10; # встановлює або керує кількість повторних запитів при помилці  snmp
    $olt->{tmpl}=&template({TYPE => $olt->{pon_type}}); # готує хеш ОІДів  snmp  для конкретного типу ПОНу (епон/жпон)
# debug Dumper($olt); Якщо розкоментувати, покаже готовий до роботи хеш ОЛТ
    $data->{olt} = &_check_olt($olt) if $olt->{status}==9; первинна ініціалізація ОЛТ (робиться 1 раз при першому скануванні нової ОЛТ)
    $data->{llid_name} = &_check_llid($olt); # сканування та генерація хешу всіх пон портів ОЛТ
# $data->{onu_unreg} = &_unregistered($olt); #не реалізовано - сканування незареєстрованих ОНУ (не рентабельно, зроблю трапами)
    $data->{onu_list} = &_monitoring($olt);# if !($step%2); #  Власне, сканування та генерація хешу всіх ОНУ на ОЛТ з типом ПОН з налаштувань
    $data->{fdb} = &_get_fdb($olt) if ( $olt->{cfg}{fdb_force_telnet} || !($step%3) ); # сканування МАКів за всіма ОНУ на ОЛТ
# debug Dumper($data); # якщо розкоментувати, покаже хеш результатів усіх опитувань
    return $data; # віддає хеш результатів всіх опитувань головному модулю для внесення до БД
}
```

Першим із функції **main** викликається генерація ОІДів із **template**

```
sub template {
    my $hash=shift;
    # example: &template({TYPE => 'epon'});
    # example: &template({TYPE => $olt->{pon_type}});
    my %snmp = (
        epon => {
            get_ports => {},
            monitor => {
                'ONU_SN' => {
                    OID => '.1.3.6.1.4.1.3902.1015.1010.1.1.1.1.1.4',
                },
                'ONU_INDEX' => {
                   # OID => '.1.3.6.1.4.1.3320.101.10.5.1.1',
                }, # bin2mac
                'ONU_STATUS' => {
                    OID => '.1.3.6.1.4.1.3902.1015.1010.1.7.4.1.17',
                },
                'ONU_TX_POWER' => {
                    OID => '.1.3.6.1.4.1.3902.1015.1010.1.1.1.29.1.4',
                    GET => 1, # для ОІДів, які не працюють за запитом snmpwalk (при get запиті в кінець ОІДА дописується
".$ONU_INDEX")
                }, # tx_power = tx_power * 0.1;
                'ONU_RX_POWER' => {
                    OID => '.1.3.6.1.4.1.3902.1015.1010.1.1.1.29.1.5',
                    GET => 1,
                }, # tx_power = tx_power * 0.1;
            },
            fdb => {},
            unworked => {}, # For history. Not used anywhere
        },
        gpon => {
            get_ports => {},
```

```
            monitor => {},
            fdb => {},
            unworked => {}, # For history. Not used anywhere
        },
    );
    return $hash->{TYPE} ? $snmp{$hash->{TYPE}} : \%snmp;
}
```

🔧 Fix Me!

Структура БД

```
CREATE TABLE IF NOT EXISTS pon_bind (
  id SMALLINT(5) UNSIGNED NOT NULL AUTO_INCREMENT,
  sn CHAR(18) NOT NULL,
  olt_id SMALLINT(4) NOT NULL,
  llid CHAR(16) NOT NULL,
  name CHAR(32) NOT NULL DEFAULT '',
  USER INT(10) NOT NULL DEFAULT '0',
  place INT(10) NOT NULL DEFAULT '0',
  tx CHAR(6) DEFAULT NULL,
  rx CHAR(6) DEFAULT NULL,
  STATUS CHAR(30) DEFAULT NULL,
  dereg CHAR(24) DEFAULT NULL,
  changed INT(11) NOT NULL DEFAULT '0',
  PRIMARY KEY (id),
  UNIQUE KEY uniq (olt_id, lid) USING BTREE
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COMMENT='ONU list';
```

```
DROP TRIGGER IF EXISTS `tr_aft_ins_pon`;
DELIMITER $$
CREATE TRIGGER `tr_aft_ins_pon` AFTER INSERT ON `pon_bind` FOR EACH ROW REPLACE INTO `pon_mon`(`bid`, `rx`, `tx`,
```

```
`time`)
 VALUES (NEW.id, NEW.rx, NEW.tx, NEW.changed)
$$
DELIMITER;

DROP TRIGGER IF EXISTS `tr_aft_upd_pon`;
DELIMITER $$
CREATE TRIGGER `tr_aft_upd_pon` AFTER UPDATE ON `pon_bind` FOR EACH ROW IF (FORMAT(NEW.rx,1) <>FORMAT(OLD.rx,1)
OR FORMAT(NEW.tx,1) <>FORMAT(OLD.tx ,1)) THEN
      INSERT INTO `pon_mon`(`bid`, `rx`, `tx`, `time`)
      VALUES (NEW.id, NEW.rx, NEW.tx, UNIX_TIMESTAMP() );
    END IF
$$
DELIMITER;
```

```
CREATE TABLE IF NOT EXISTS pon_fdb (
  mac CHAR(18) NOT NULL,
  vlan SMALLINT(4) NOT NULL,
  llid CHAR(16) NOT NULL,
  olt_id SMALLINT(4) NOT NULL,
  name VARCHAR(32) DEFAULT NULL,
  uid INT(10) DEFAULT NULL,
  ip CHAR(16) DEFAULT NULL,
  TIME INT(11) NOT NULL DEFAULT '0',
  UNIQUE KEY mac (mac) USING BTREE,
  KEY llid (llid),
  KEY name (name) USING BTREE,
  KEY ip (ip) USING BTREE
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COMMENT='ONU list';
```

```
CREATE TABLE IF NOT EXISTS pon_mon (
  bid mediumint(7) NOT NULL,
  tx CHAR(6) DEFAULT NULL,
  rx CHAR(6) DEFAULT NULL,
```

```
  TIME INT(11) NOT NULL DEFAULT '0',
  KEY TIME (TIME)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COMMENT='tmp ONU monitoring';
```

```
CREATE TABLE IF NOT EXISTS pon_olt (
  id SMALLINT(5) UNSIGNED NOT NULL AUTO_INCREMENT,
  name VARCHAR(30) NOT NULL DEFAULT 'new olt',
  ip VARCHAR(15) NOT NULL,
  snmp_port SMALLINT(6) NOT NULL DEFAULT '0',
  vendor VARCHAR(20) NOT NULL DEFAULT '',
  model VARCHAR(20) NOT NULL DEFAULT '',
  firmware VARCHAR(17) NOT NULL DEFAULT '',
  ro_comunity VARCHAR(32) NOT NULL DEFAULT '',
  rw_comunity VARCHAR(32) NOT NULL DEFAULT '',
  pon_type CHAR(8) NOT NULL DEFAULT 'gpon',
  mng_tmpl INT(4) NOT NULL DEFAULT '0',
  param VARCHAR(1024) DEFAULT NULL,
  descr VARCHAR(250) NOT NULL DEFAULT '',
  STATUS tinyint(1) NOT NULL DEFAULT '9',
  enable tinyint(1) NOT NULL DEFAULT '0',
  changed INT(11) NOT NULL DEFAULT '0',
  PRIMARY KEY (id),
  UNIQUE KEY nas_identifier (ip, snmp_port, pon_type) USING BTREE
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COMMENT='OLT list';
```

```
CREATE TABLE IF NOT EXISTS pon_onu (
  id mediumint(7) UNSIGNED NOT NULL AUTO_INCREMENT,
  sn CHAR(32) NOT NULL,
  vendor CHAR(64) NOT NULL DEFAULT '',
  model CHAR(24) NOT NULL DEFAULT '',
  firmware CHAR(24) NOT NULL DEFAULT '',
  descr CHAR(255) NOT NULL DEFAULT '',
  changed INT(11) NOT NULL DEFAULT '0',
  PRIMARY KEY (id),
```

```
  UNIQUE KEY sn (sn)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COMMENT='ONU list';
```

```
CREATE TABLE IF NOT EXISTS pon_ports (
  id SMALLINT(5) UNSIGNED NOT NULL AUTO_INCREMENT,
  olt_id SMALLINT(11) NOT NULL,
  name VARCHAR(30) NOT NULL DEFAULT 'new olt',
  p_index VARCHAR(30) NOT NULL,
  TYPE VARCHAR(30) NOT NULL DEFAULT '',
  shelf tinyint(2) NOT NULL DEFAULT '0',
  slot tinyint(2) NOT NULL DEFAULT '0',
  port tinyint(2) NOT NULL DEFAULT '0',
  tx VARCHAR(10) NOT NULL DEFAULT '',
  descr VARCHAR(250) NOT NULL DEFAULT '',
  STATUS tinyint(1) NOT NULL DEFAULT '9',
  enable tinyint(1) NOT NULL DEFAULT '0',
  changed INT(11) NOT NULL DEFAULT '0',
  PRIMARY KEY (id),
  UNIQUE KEY olt_id (olt_id, p_index)
) ENGINE=InnoDB DEFAULT CHARSET=utf8 COMMENT='OLT ports';
```

Приклад таблиць графіків сигналів

```
DROP TABLE IF EXISTS z2021_01_01_pon;
CREATE TABLE IF NOT EXISTS z2021_01_01_pon (
  bid mediumint(7) NOT NULL,
  tx CHAR(6) DEFAULT NULL,
  rx CHAR(6) DEFAULT NULL,
  TIME INT(11) NOT NULL DEFAULT '0',
  KEY TIME (TIME)
) ENGINE = InnoDB DEFAULT CHARSET = utf8;
```