

OpenWrt (Beesip) And Zabbix

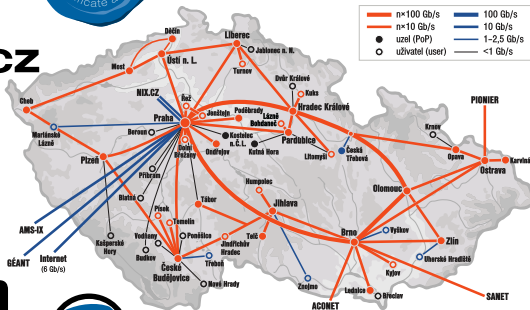
Lukas Macura



September 12, 2015



About CESNET



About Silesian University



Silesian University in Opava

- Faculty of Philosophy and Science in Opava (abbreviated FPF)
- *School of Business Administration in Karviná (abbreviated OPF)*
- *Faculty of Public Policy in Opava (abbreviated FVP)*
- *Institute of Mathematics in Opava (abbreviated MU)*



Contents

- 1 BEESIP
- 2 OpenWRT
- 3 Zabbix In OpenWRT
- 4 UCIPROV
- 5 Showcase - EduroamAP
- 6 Examples

About Beesip

- Bright Efficient Embedded Solution for IP Telephony

<http://beesip.cesnet.cz>

- OpenWRT based
- Bleeding Edge

We make glue of latest VoIP packages of telephony feed. Everything in one, downloadable image.

- Easy to use

Just download image and use your favourite virtualisation technology like KVM or VmWare

- Scalable

Beesip (like OpenWrt) can run on small home router for small usage, on Raspberry PI or as virtual appliance for high speed

- Easy to port

Beesip can run almost on any HW which is supported by OpenWrt



About Beesip

- **Configurable**

Beesip is highly configurable during build process. You can configure any target by your own (add files, deploy certificates into image, disable or enable packages)

- **Manageable**

Beesip is not like normal operating system with entire file-system writeable. Configuration is on separate place. This means that you can upgrade Beesip and use same config.

- **Distributed as Image**

Just download, flash and go. If HW is broken, change HW and copy only configuration.

- **Provisioning**

Beesip can be provisioned using uciprov package. You can run lot of images with common configurations, stored on external place (like http,tftp,https,...).



Another uses of Beesip

- Embedded monitoring probes
- Security probes
- Appliances distributed as image
- All cases where many devices needs to be configured, monitored and managed from one place

About OpenWRT

- Linux distribution for embedded devices
- Precise build system with many options
- OPKG packaging system
- Big effort on size
- Specialised versions of software (like netifd, ubusd, ...)
- Many supported devices
- Everything is cross-compiled during build

About OpenWRT packages

- Each package is set of makefiles, config and (optionally) patches
- Many times not so easy to port
- Result is OPKG package
- Can be embedded into image or be part of repository
- Everything is highly configurable (even package options)

OpenWRT UCI

- Unified Configuration Interface
- OpenWRT core uses UCI natively
- It is set of config files with common structure
- Package can be native UCI, UCIfied or non-UCI

Example (interface config)

```
config 'interface' 'wan'
    option 'proto' 'dhcp'
    option 'ifname' 'eth0.1'
    list   'dns'    '192.168.1.1'
    list   'dns'    '192.168.1.2'
```

Zabbix packages for OpenWRT

Official repository

- Only non-UCI version and passive template

Our repository

- Full UCI version
- Easy to extend (userparams)
- Extended template
- Zabbix-UCI is part of Beesip image
- Init script parse UCI config and create configs
- Auto-discovery is working (interfaces, wifi interfaces, wifi channels, wifi clients, packages, ...)
- It is easy to make new discovery working (shell library)

Zabbix server for OpenWRT

- It is not common to use server in embedded environment
- Only server part is working, PHP frontend is too robust for OpenWRT
- It uses only SQLITE database
- It works but it is not manageable (missing frontend, problems with locking)

Example server config

```
config zabbix_server config
option enable true
option dbname '/tmp/zabbix_server.sqlite3'
```

Zabbix proxy for OpenWRT

- It is possible to run in embedded environment
- It uses only SQLITE database
- SQLITE DB can be in RAM (/tmp)
- Does not work on very small devices (RAM and CPU problems)
- Can be used as light efficient proxy for SoHo
- Works for me on TP-LINK TL-WDR3600/4300/4310 and external storage
- Can be built as Vmware appliance

Example proxy config

```
config zabbix_proxy config
option enable true

#option proxymode 0
#option server 127.0.0.1
option dbname '/tmp/zabbix_proxy.sqlite3'
# If not set, system.hostname is used
#option hostname somehostname
```

Zabbix agent for OpenWRT

- Can be extended by zabbix-extra-wifi, zabbix-extra-network, zabbix-extra-mac80211 packages
- Works even on very small devices
- Can collect very useful informations (wifi, network, processes, ...)

Example agent config

```
config zabbix_agentd config
option enable 1

option server 127.0.0.1
#option serveractive zabbix
#option listenport 10050
#option hostname somehostname
option hostnameitem system.hostname
option allowroot 1
#option disableactive 0
#option disablepassive 0
option logremotecommands 1
#option enableremotecommands 0
```

Zabbix-UCI User parameters

Example userparams config

```
config userparm
    option key          owrt.ipackages
    option internal      0
    option cmd           'opkg list-installed | cut -d " " -f 1 | tr "\n" " "'
    option lock opkg

config userparm
    option key          owrt.apackages
    option internal      0
    option cmd           'opkg list | cut -d " " -f 1 | tr "\n" " "'
```

UCIPROV basics

UCI provisioning system is tool for OpenWrt distribution, which will allow OpenWrt box to be automatically configured via network. Most common usage is to download UCI configuration to the box just after factory defaults. In conjunction with Zabbix auto-discovery, it is powerful tool

- UCIPROV is reason why Zabbix packages should be UCIified
- OpenWRT package for provisioning
- Primarily used for UCI provisioning
- It is modular
- More ways how to get provisioning URL

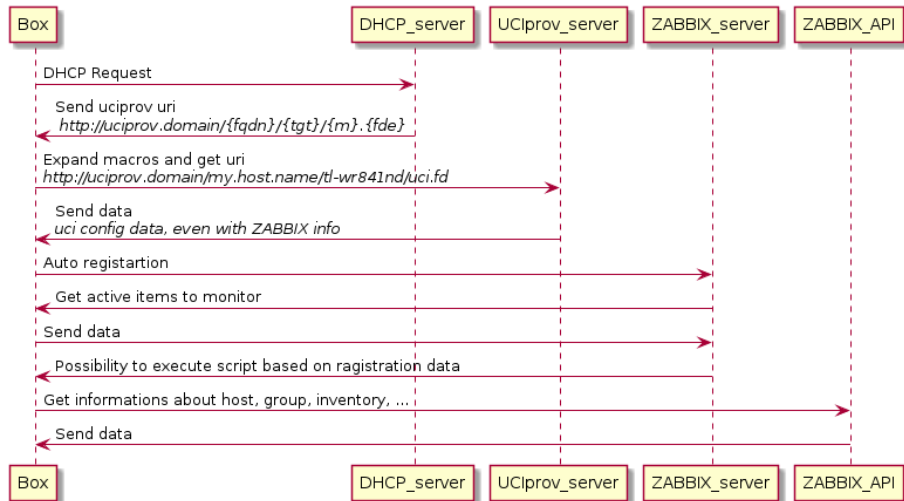
UCIPROV modules

List of some modules for UCIPROV. Easy to add next (shell scripts)

- tgz - for deploying tar.gz files over root fs
- opkg - for installing packages
- mgmt - to setup management (deploy ssh keys, disable password auth, ...)
- services - to start/stop/disable/enable any service
- ssldeploy - to deploy ssl certificates
- zabbix - to communicate via ZABBIX Api
- recovery - to put device back to factory defaults



UCIPROV flow



UCIPROV usage

- Device boots
- It gets informations about URIs and macros
- It fetch informations from URIS expanding macros
- It runs module depending on fetched content

Example

- URI `http://uciprov.domain/{fqdn}/{tgt}/{m}.{fde}` expands to `http://uciprov.domain/my.host.name/tl-wr841nd/uci.fd`
- Get content
- Apply UCI config
- Get `http://uciprov.domain/my.host.name/tl-wr841nd/tgz.fd`
- Untar tgz file into root
- Get `http://uciprov.domain/my.host.name/tl-wr841nd/services.fd`
- Runs or stop services

UCIPROV And Zabbix

- Together with Zabbix and auto-discovery there are more possibilities
- Device uses UCIPROV to preconfigure Zabbix agent
- Next, it is auto-registered to Zabbix server
- It is automatically added and monitored or has to be manually approved for security
- It can fetch information using Zabbix API
- It can update information in Zabbix
- Zabbix can use trigger and external commands to automatize
- All devices under common management and control
- In future, device could be managed directly via Zabbix



UCIPROV macros

```
{hostname}      # Hostname
{domain}        # Domain from DHCP
{dns}           # DNS server IP (first of list)
{fqdn}          # Fqdn of host
{mac}           # Mac address of provisioning interface (aa-bb-cc-dd-ee-ff)
{mac2}          # Mac address of provisioning interface (aabbccddeeff)
{mac3}          # Mac address of provisioning interface (aa:bb:cc:dd:ee:ff)
{ip}            # IP address of provisioning interface
{fd}            # Box in factory defaults (0/1)
{fde}           # Expands to 'fd' or 'nofd' if box is or is not in factorydefaults
{ifname}        # Provisioning interface name (like eth0)
{oifname}       # Owrtn Provisioning interface name (like wan)
{sr}            # Revision of distribution (like git rev)
{cn}            # Codename of distribution (like barrier_breaker)
{rl}            # Release of distribution (like 14.07)
{tgt}          # Distribution target and subtarget (like x86_64/generic)
{cpu}           # CPU (like x86_64)
{sub}           # Subtarget
{board}         # Board identification string. Like tplink-wdr3600-v1
{nettime}       # Network time obtained (0/1)
{m}             # Module which performs given action (eg. uci for uciprov, tgz for tgz module)
{boxid}         # Persistent hash for this box.
```

- Beesip(OpenWRT) based project for managing Eduroam APs
- It uses common OpenWRT HW to make Eduroam wireless network
- It can be used to create any wireless network
- All devices are configured from provisioning server
- Common configs are shared
- Device specific configs in separate files
- It is enough to flash device and it will be configured automatically
- Replacement of device is easy (config is on provisioning server)

EduroamAP - provisioning structure

Filesystem

```
/default/ # Defaults for all devices
/default/uci.fd # UCI config for all devices
# Eg. zabbix server location, common network settings, ...

/tl-wr841nd-v7/ # Defaults for this type of devices
/tl-wr841nd-v7/uci.fd # Eg. switch and port configuration
/tl-wr841nd-v7/sysupgrade.fd # Sysupgrade image

/aa-bb-cc-dd-ee-ff/ # Device specific config
/aa-bb-cc-dd-ee-ff/uci.fd # Eg. wifi channels settings
```

EduroamAP - list of auto-discovered hosts

Automatically deployed devices

Device which boots and is provisioned is automatically added to Zabbix server.

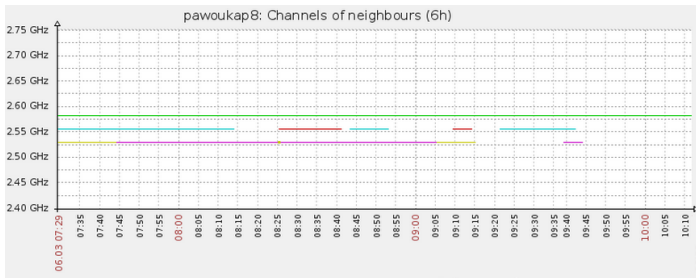
Name	Interface	Status 	Availability
pawoukap8	192.168.2.27: 10050	Enabled	   
pawoukap7	192.168.2.26: 10050	Enabled	   
pawoukap6	192.168.2.25: 10050	Enabled	   
pawoukap5	192.168.2.24: 10050	Enabled	   
pawoukap2	192.168.2.21: 10050	Enabled	   
pawoukap4	192.168.2.23: 10050	Enabled	   
pawoukap1	192.168.2.20: 10050	Enabled	   
pawoukap3	192.168.2.22: 10050	Enabled	   



EduroamAP - channels of neighbour devices

Channels auto-discovery

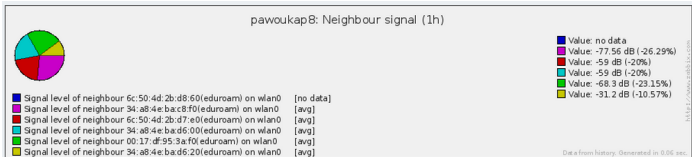
Wifi channels are auto-discovered and monitored. Zabbix monitors all neighbours and can use trigger to alert new device or device with high noise



EduroamAP - neighbour devices signal level

Neighbours signal level

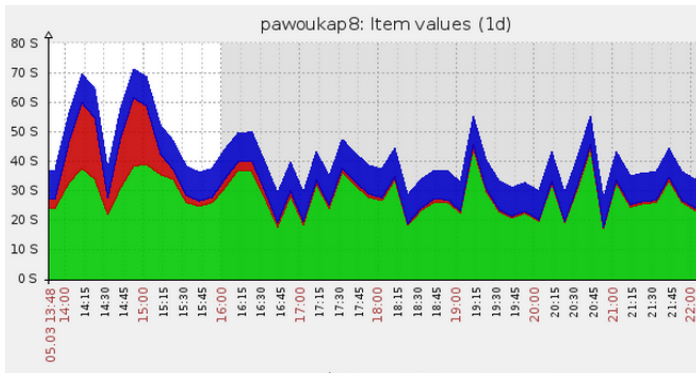
Each neighbour is monitored automatically after discovery



EduroamAP - channel times

Channel times

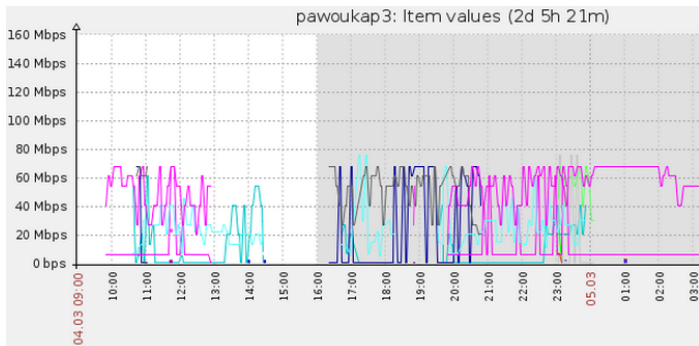
If supported by HW, it is possible to report times on each channel (busy, active, transmit and receive). If values are bad, Zabbix can start trigger or action



EduroamAP - client speeds

Client speeds

Each client connected to AP for longer time is auto-discovered. Next, it is possible to monitor signal, speed and noise levels



Thank you for attention

Lukas Macura

`lukas.macura@cesnet.cz`

Jiri Slachta

`jiri.slachta@cesnet.cz`



Example - Creating JSON in OpenWRT

- Easy to create any kind of discovery
- It uses internal OpenWRT JSON shell parser
- Output of any command can be feed to 'discovery_stdin'
- Output is JSON for auto-discovery
- No dependency - pure shell code

Example code for packages auto-discovery

```
owrt_packagediscovery(){  
    local pkg version description  
  
    opkg list-installed | \  
        discovery_stdin "${#PACKAGE}" "${#VERSION}"
```

Example result

```
[ { "${#PACKAGE}": "acpid", "${#VERSION}": "2.0.22-1" },  
  { "${#PACKAGE}": "alsa-lib", "${#VERSION}": "1.0.27.2-1" },  
  { "${#PACKAGE}": "asterisk11", "${#VERSION}": "11.15.0-2" },  
  { "${#PACKAGE}": "asterisk11-app-chanisavail", "${#VERSION}": "11.15.0-2" },  
  { "${#PACKAGE}": "asterisk11-app-chanspy", "${#VERSION}": "11.15.0-2" }  
]
```

Example - Parsing JSON in OpenWRT

- Easy to communicate with Zabbix API via shell
- It uses internal OpenWRT JSON shell parser
- Predefined functions for login and get info about host
- Need more work
- No dependency - pure shell code

Example code to get inventory data for given host using Zabbix API

```
fetch_data_from_api(){
    zabbix_api_login "http://zabbix.fqdn/json_rpc.php" user password

    hostname=$(hostname)
    json_load "${zabbix_get_host $hostname}"
    json_get_keys keys result
    json_select result
    json_select $keys
    json_get_keys keys inventory
    json_select inventory
    for i in $keys; do
        json_get_var var "$i" && [ -n "$var" ] && setmacro "$i" "$var"
    done
}
```