European Collaborative Innovation Centres for Broadband Media Service

Enabling individual experiments in KAUMesh

Case Study Final Report

EUROPEAN UNION
European Regional Development Fund

Investing in the future by working together for a sustainable and competitive region

The Interreg IVB North Sea Region Programme

Enabling individual experiments in KAUMesh

Peter Dely, Ankit Bhatnagar, Andreas J. Kassler

Abstract
This report documents the changes done to the KAUMESH scheduler, in order to incorporate ability for the nodes to boot different images for allowing more flexibility in experimentation on KAUMESH.

Index Terms
Performance Evaluation, Testbed, Burst Transmission

1 Introduction

KAUMesh is an experimental Wireless Broadband Mesh Network based on 802.11a/b/g WLAN based devices (the mesh node) that has been deployed at the Karlstad University Campus. KAUMesh comprises currently 20 mesh nodes, which are permanently installed to cover large areas inside the House 21. Researchers and students from Karlstad University and partner organisations use KAUMesh to develop and evaluate next-generation WLAN and Wireless Mesh solutions.

Currently the KAUMesh testbed provides the following features:

- 3 IEEE 802.11a/b/g wireless cards per node
- Each node has an Ethernet connection for control and logging
- Remote reset of nodes
- Integrated node reservation and access control system
- Graphical monitoring tools
- A wide range of routing protocols and traffic generators

KAUMesh offers a flexible architecture, both in terms of hardware and software. The deployment allows a quick re-arrangement of the mesh nodes. Users can control almost all aspects of the KAUMesh software (including the OS-kernel) remotely. For monitoring and control purposes each mesh node has a wired Ethernet connection to a managed network in order to send monitoring data to a monitoring server, located in the fixed network. The monitoring server also for hosts configuration information for the mesh nodes and provides software via TFTP and NFS. One Ethernet interface using the network prefix 192.168.31.0/24 is used to connect to the Management/Monitoring server. The wireless interfaces use the network prefix 192.168.30-28.0/24

2 Boot Process for KAUMesh Nodes

In KAUMESH nodes, boot process is controlled by REDBOOT firmware. The boot process on the nodes can be configured by using the command “fconfig” on REDBOOT command line interface which can only be entered through console on nodes. For connecting to the console, one can use serial port and cable. The REDBOOT environment allows to load files using TFTP(trivial FTP) and other protocols also. KAUMESH nodes use TFTP for boot up. The files are present on the KAUMESH server which has a TFTP server running. The KAUMESH server also serves as DHCP server for the nodes, BOOTP requests from nodes are mapped to their IP addresses using the MAC addresses.

In the original deployment, all the nodes are configured to boot up from the same image i.e. they all load same files from the server. The boot script was as follows:

```
load -r -v -b 0x01600000 zImage
load -r -v -b 0x00800000 ramdisk.gz
exec -c "console=ttyS0,115200 root=/dev/ram0 initrd=0x00800000" 0x01600000
```

This boot script directs the nodes to load files "zImage" and "ramdisk.gz" from the TFTP server (installed on default server-KAUMESH server in our case) using default protocol TFTP. The last line executes the kernel loaded in the main memory of the node. The TFTP server is configured to look up files in the directory "/tftpboot" on the KAUMESH server. The files "/tftpboot/zImage" and "/tftpboot/ramdisk.gz" must be present on the KAUMESH server (configured as default server in REDBOOT for KAUMESH nodes).

The main problem with this system is that each node boots the same image with the same configuration and hard/software. Therefore, we cannot easily perform many experiments in parallel which requires support for different boot up images. Also, in the old Boot system, if one wants to do an experiment with a specific protocol, the user has to change the files "/tftpboot/zImage" and "/tftpboot/ramdisk.gz". This change has been automated in the new system.

In the new Boot system, each of the node has been reconfigured to boot up different images. This has been done by changing the boot script on each of the nodes to load from their specific files in the "/tftpboot " directory.

e.g. the new boot script for node 8 is:

```
load -r -v -b 0x01600000 zImage8
load -r -v -b 0x00800000 ramdisk8.gz
exec -c "console=ttyS0,115200 root=/dev/ram0 initrd=0x00800000" 0x01600000
```

In the files "/tftpboot/zImage8" and "/tftpboot/ramdisk8.gz" are symbolic links to the files for the image chosen by the user booking the "Node #8". The symbolic links are created based on the reservations made (the BOOT image chosen) by the user using the booking system for the nodes. They are created by phpscript "/sbin/change_images.php" on KAUMESH server. The cron daemon periodically (every 15 minutes) executes the script "/sbin/change_images.php", which creates symbolic links for the current reservation in the directory "/tftpboot" after looking up the scheduler database.

3 Node Reservation System

The main purpose behind the changes done to the scheduler has been to allow more flexibility in experiments done on KAUMESH. The new system will allow integrated experiments over wireless networks and Internet. Another objective is to allow parallel experimentation i.e. different experiments at the same time requiring different boot images. This requires a node reservation system in place which allows reserving nodes and frequencies for different experiments at selected dates. The reservation system is implemented by a set of PHP scripts on the KAUMesh server, which allow different users to specify what nodes they want to book for what time and what boot images to use.

The following features are supported:
1. Boot Image Selection: The new system allows users to select boot image for a node from a set of available images as well to boot from their own compiled images (Custom) for a specific node.

2. Boot Image Manipulation: The new system automatically puts the appropriate links in place for the reservation, so that the selected node boots a specific image.

3. Automatic Restart: The new system restarts a node when a reservation is starting in time, so that the appropriate image boots up the node.

Link Creation and Restart
We have created a file change_images.php to perform the Symbolic link creation (from files /tftpboot/zImage<node_id> to file in repository or to the custom image) and restarting nodes with new reservations. This file is located as "/sbin/change_images.php" on KAUMESH server.

This file first deletes the "/tftpboot" directory so as to remove old links.

Then it finds out the current reservations for the nodes by doing comparisons on fields start date, end date, start time and end time with current date and current time (see get_curr_reservation() in the file /var/www/schedule/lib/). For each of the current reservation, soft links are created from /tftpboot/zImage<node_id>, /tftpboot/ramdisk<node_id>.gz to link_zImage and link_ramdisk respectively in the reservation entry.

For the current reservations which have started less than 15 minutes before (the New reservations), the restart is performed by calling the script /srv/meshhome/power/restart-n<node_id>.sh .

The link creation and restart is performed every 15 minutes as a cron job (use command "sudo crontab -e" on KAUMESH server) by calling the command "php /sbin/change_images.php").

Changes done on nodes
All the KAUMESH nodes' REDBOOT environment has been reconfigured. Only the boot script has been changed using the command "fconfig" on the redboot command prompt.

The new boot script directs the nodes to boot from the files "/tftpboot/zImage<node_id>, /tftpboot/ramdisk<node_id>.gz". It is as follows:

```
load -r -v -b 0x01600000 zImage<node_id>
load -r -v -b 0x00800000 ramdisk<node_id>.gz
exec -c "console=ttyS0,115200 root=/dev/ram0 initrd=0x00800000" 0x01600000
```

e.g. the new boot configuration for node 8 is (by replacing <node_id> by 8) :

```
load -r -v -b 0x01600000 zImage8
load -r -v -b 0x00800000 ramdisk8.gz
exec -c "console=ttyS0,115200 root=/dev/ram0 initrd=0x00800000" 0x01600000
```

Changes done to the Booking System
The booking system is implemented through a set of PHP scripts. In order to support the specification of different and customized boot images, the following changes have been applied:
The files which have been changed on the scheduler in the directory "/var/www/schedule/" are:

lib/Reservation.class.php - This file implements class for capturing reservation data. It has been changed to include the selected image parameters.

lib/db/ResDB.class.php - This file implements add_res() and mod_res() methods for adding new reservations and modifying existing reservations respectively. It has been modified to include boot image parameters so they are entered/updated in the table "reservations".

templates/cpanel.template.php - This file displays the days' reservation for the user. It has been modified so that the image selected by the user is also displayed.

templates/reserve.template.php - This file essentially contains all content displayed on the reservation page. So, it has been modified to include Image selection drop-down menu and the image description box.

reserve.php - modified to get image parameters from form data.

functions.js - This file has all the javascript functions. The check() function has been modified to give alerts when no image is selected.

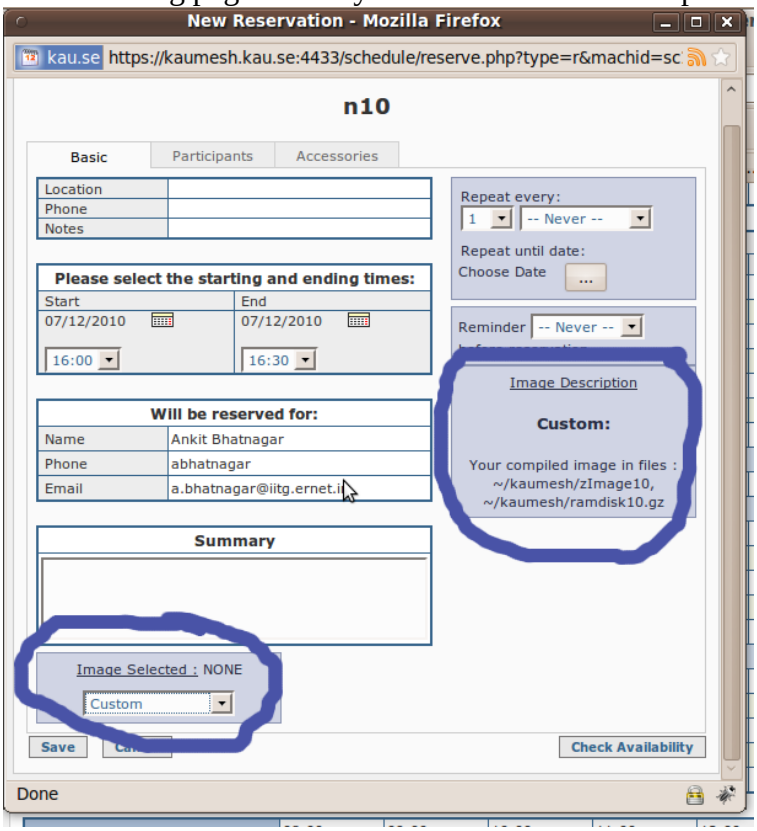
0.3.3.2 New Files

/sbin/change_images.php - The file which performs the boot Image manipulation.

4 Using the system

Booking a Node for an experiment

The following page allows you to select nodes for experiments:



The reserve.php script allows you to book nodes for experiments at a given time. Selecting an Image is compulsory for all the users while they are doing reservations. Without doing the proper reservations, the boot image links would not be in place and the nodes will not boot up.
The new system also allows you to select a Custom Boot Image for each node. This allows users to have a different boot image for each of the nodes he has reserved. For the Custom Image, the boot image must be present in the following files

"/home/<username>/kaumesh/zImage<node_id>" and
"/home/<username>/kaumesh/ramdisk<node_id.gz>"

on the KAUMESH SERVER (kaumesh.kau.se)

replace <username> by your username on KAUMESH SERVER and <node_id> by the node you are using. For example, if e.g. the user with the username abhatnagar is reserving node 10 for Custom boot up, the given custom made boot image must be present in the files:

```
/home/abhatnagar/kaumesh/zImage10
/home/abhatnagar/kaumesh/ramdisk10.gz
```

on KAUMESH SERVER (kaumesh.kau.se)

The symbolic links are created in the "/tftpboot" directory of the KAUMESH server by the file "/sbin/change_images.php". This file is executed every 15 minutes on the server by the cron daemon. This is needed because the scheduler allows reservation for a minimum period of 30 minutes. Thus, it is required that the symbolic links are revised at least every 30 minutes, but we have kept the period 15 minutes to include changes for possible updates on the reservation. So if one is rebooting a node after changes to the boot image, one needs to wait for the next execution of this script. For example, if you modify a reservation for node 8 to boot from OLSR instead of Default at 9:35 am, you must wait till 9:47 am for the right links to be put in place. The 2 min delay is caused because in case all the reservations are new, all the nodes will be rebooted after placing the links, which causes a delay of 5 second for each of the node. If there are scheduling delays, the time required may be more.

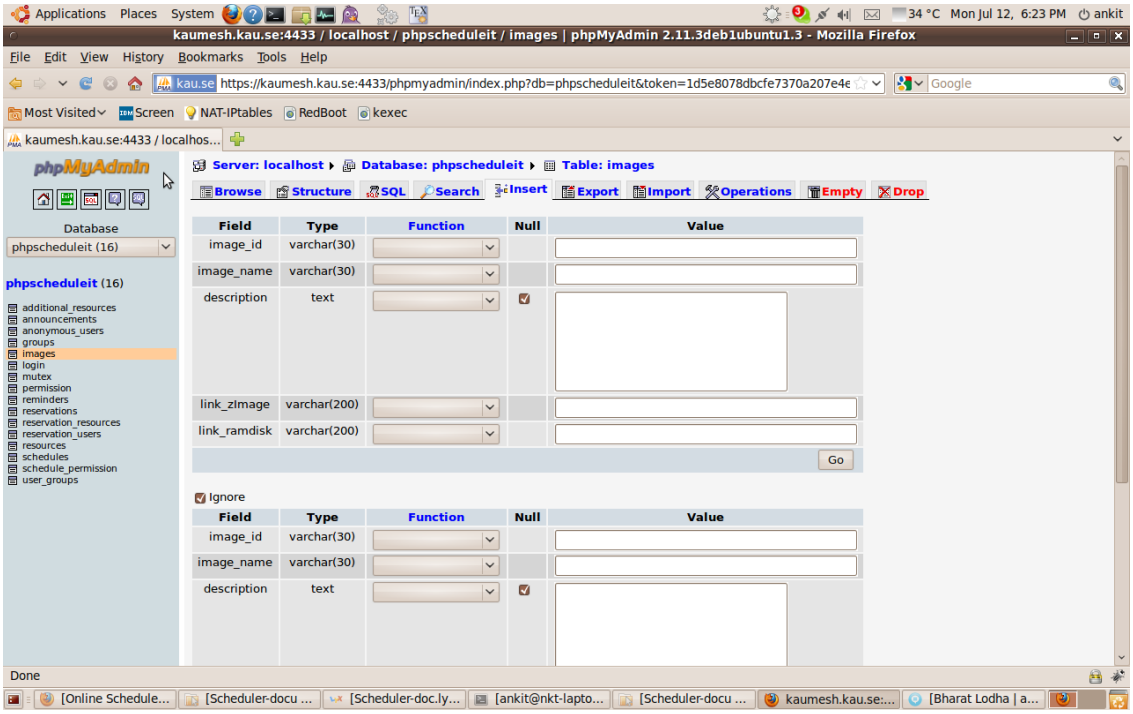
Adding a New BOOT image OR Updating an existing BOOT Image

Predefined Boot images allow the possibilities for experimenters to select among customized images. For example, one can define a Boot image called "AODV-MR" having support for AODV and multichannel multiradio. Another Boot image called "OLSR" could be compiled for single radio using OLSR, etc. Experimenters then just need to pick the right pre-compiled boot image from a list and boot up a set of nodes with the selected images to perform experiments. In this section, we explain, how the booking system can be customized to define new boot images or updating existing ones so as to appear correctly in the user interface for booking a node.

In this section, we explain how a new choice of boot image can be made available in the drop-down menu of choices in the booking system for the ndodes for the users in the scheduler. This may be needed, when a Boot image with some new functionality is to be made available to all the users of the KAUMESH. For doing this, the administrators must login into https://kaumesh.kau.se:4433/phpmyadmin/phpmyadmin at KAUMESH server. The username and password can be found in the config file of the scheduler. You must select the database "phpscheduleit" when you login. In the database, you need to adjust the table "images".

When adding a new Image, the zImage and ramdisk.gz files must be present for that image in the repository folder. Currently, the repository folder is "/repo". This can be changed to a new directory by changing the constant DIR_REPO in the file "/var/www/schedule/config/constants.php".

For example if you want to add a new image AODB with image files "zImage-aodb, ramdisk.gz-aodb", then you must copy these files in the repository folder (currently "/repo"). Once you have files "/repo/zImage-aodb" and "/repo/ramdisk.gz-aodb" in place, you need to do the database entry. For this after logging in, follow the links phpscheduleit->images->insert. You will get a window for entering new Image:



In this window, you can do the entry for the new Image you require.

image_id is primary key for the table, so it must be unique(it is recommended that you give image_id in increasing order, e.g. if currently the maximum value of image_id is 3, then you should give new entry image_id as 4)- It must also be an integer.

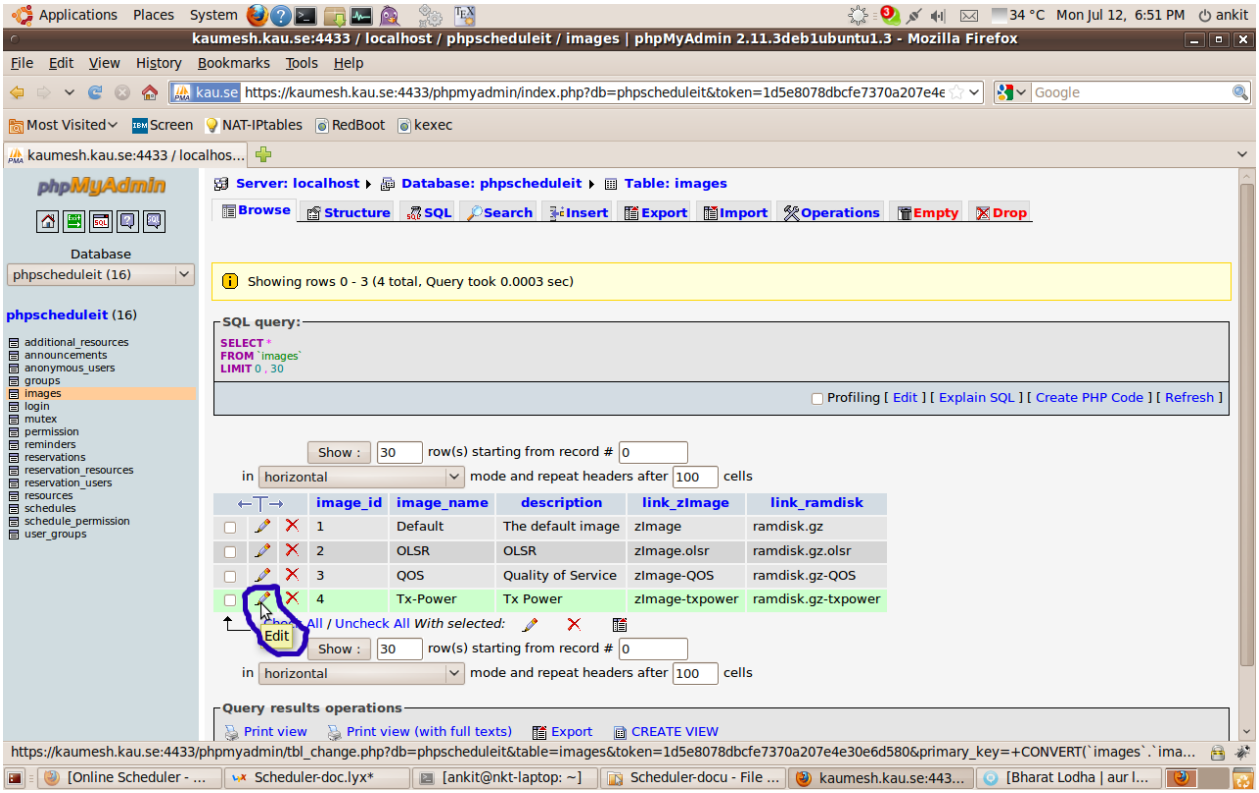
all other fields must not be left empty- image_name and description must be entered appropriately

link_zImage is the name of zImage file for the Boot image in the repository folder, e.g. for the previous situation, this should be zImage-aodb

link_ramdisk is the name of ramdisk file for the Boot image in the repository folder, e.g. for the previous situation, this should be ramdisk.gz-aodb

Once you enter the data, just click on Go.

For updating an existing image, after logging in, follow the links phpscheduleit->images->browse. The following screen will appear:



Here, we can see that the following 4 images are available: The default image, an image called OLSR, an image called QOS and an image called Tx-Power.

To edit an Image, just click on the edit link for that entry, you will get a new window. Update the columns you want to change for the selected image and click on Go.

5 Conclusion

The presented updates allow experimenters to define and upload specific customized images for the KAUMesh nodes to boot from. This allows the support of different experiments using different Kernels, protocols, etc.. The main disadvantage of the new system is that it is too strict. It adheres to the reservations placed in the scheduler and users can boot up only the nodes for which booking has been done. One can obviously surpass the scheduler and boot nodes by making changes to the firewall and placing appropriate files in the folder redirected.

The future work on this system can be to add the Boot Image ADD/UPDATE system to the administrator's interface, instead of going in through the PhpMyAdmin. Another enhancement would be to allow users to upload files for custom image through the scheduler. An important feature is knowing what was the time when the script “/sbin/change_images.php” was last executed. This needs to be added to the new system.