OpenBSD as house alarm system

OpenBSD is not only for Network related projects

Fosdem 2018

Email: vincent.delft@easytransitions-ict.be

Blog: http://vincentdelft.be

Company: http://easytransitions-ict.be

Topics for today

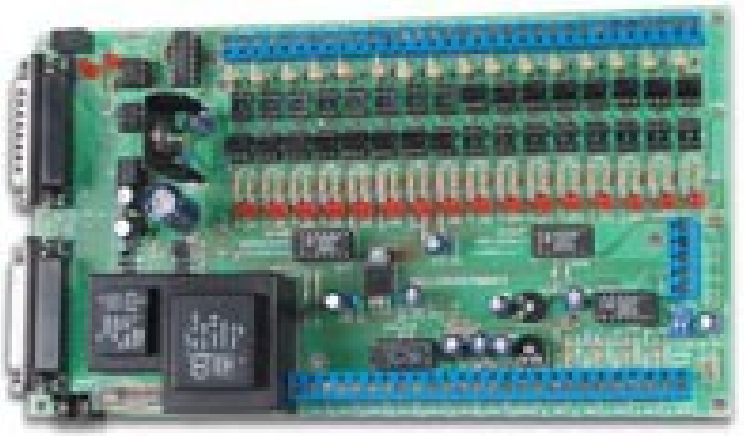
- Components (hw, sw, coding, configs)
- Lessons learned since 1998
- Problems with linux
- Why openbsd ?
- New system based on Denkovi
- Putting OpenBSD Read-only
- Conclusions

Monitor my house

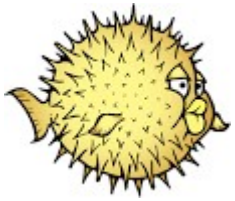
- The goals are :
  - To monitor the whole house via PIR captors with the concept of zones
  - Start a siren in case of intrusion
  - Send message (SMS) in case of intrusion
  - Be able to react on my SMS (status, activate, de-activate, ...)
  - Have a web interface in order to check the logs
  - Have a webcam in order to check the main entrances
  - Cheap solution

Hardwares

- I’ve started this project with a Velleman kit : K8000.



- An GSM Modem to send/receive SMS
- A cheap Atom 330 board with serial and parallel ports: 4GB Ram, 1.6Ghz, no disk.
- A powerful OS



Softwares

- For K8000 is has to be i386 (until OpenBSD 6.0)
- Httpd
- FTPd
- C compiler
- Python2

Programs

- Develop a C program to loop around « inb » and « oub » API to check the status of the 16 IO ports of the K8000 (i386 !!).
- Program loop around the IO ports and react based on the IO status
- But ...

SMS

- pkg_add smstools3
- Remove the pin code at SIM when booting
- Config is like this:

```
[GSM1]
device = /dev/cua00
incoming = yes
cs_convert = yes
init = AT+CHUP
eventhandler = /home/vi/smsd_event.py
```

SMS

- Define an event handler to treat the incoming SMS:

```
<prog> RECEIVED /path/to/SMSfile


      From: 32475xxxxxx
      From_TOA: 91 international,
      ISDN/telephone
      From_SMSC: 324xxxxxxx
      Sent: 17-11-02 15:29:51
      Received: 17-11-02 15:30:03
      Subject: GSM1
      Modem: GSM1
      IMSI: 206106500xxxxx
      IMEI: 350301410xxxxx
      Report: no
      Alphabet: ISO
      Length: 6

      Status
```

SMS

- To send an SMS you just have to write a file in `/var/spool/sms/outgoing/<file>`

```
To: 32475xxxxxx

The status is OK
```

Webserver

- Must be light, easy to use, ...
- At beginning it was fapws (fapws.org). Light python wsgi web server.
- Now I’m using openbsd-httpd with simple cgi scripts
- Fapws was 9KB memory, httpd is 2x3KB. But httpd is maintained by the openBSD community :-).
- Web server run in a chroot environment where only the logs and webcam images reside.
- This interface is not able to configure the alarm !!!
- Dynamic IP manage via freedns.afraid.org and cron ftp every 5 minutes

Webcam

- Old Axis camera found in ebay
- Send pictures via FTP. Use motion detection
- OpenBSD machine configured with ftpd:
- `/etc/rc.conf.local` must contain this: `ftpd_flags="-llUSA"` (for logging purposes)
- Create a valid user: `useradd` …
- Add this user in `/etc/ftpchroot`.
- Inform your camera that they can use this ftp-server / user

Lessons learned since 1998

- The time spent after each upgrade is not to under-estimate, surely with Linux distros I had
- Replace the main board every 6 or 7 years
- Always have a backup Power supply.
- Do not under estimate the different power cuts (storm, technicians, ...)
- Every year perform a cleaning of the IR captors
- Be ready to manage false alarms (cats, insects, heat systems, ...).
- Advise your neighbors (loud sirens)
- Foresee a remote access in order to manage problems
- For your bills, prefer recent low consumption board to old hardware.

Problems with linux

- I’ve run slackware (4y), redhat (3y), then gentoo (3y) between 1998 and 2007
- Several problems:
  - upgrade problems: pkg names, ...
  - Rules: lot of dependencies, ...
  - Disks crashes (ext2)

Advantages of OpenBSD

- No more disk crashes !!!. the disk was spinning between 2009 and until 2016 24hx7.
- Upgrades are really easy with OpenBSD (but I’m doing the un-recommanded upgrade process because I do not have easy access to the serial console)
- No more C program updates, but … .
- In OpenBSD, features are mature
- Secure by default.
- Very good man pages

(dis)Advantages

- But …

- Ted Unangst remove hopes to have ioperm, … for amd64 on may 2013.
- “inb”, “outb” and “ioperm” removed from OpenBSD 6.0 (2016)

```
CVSROOT:        /cvs
Module name:    src
Changes by:     guenther@cvs.openbsd.org        2016/03/23 22:56:08
Modified files:
        lib/libarch/i386: Makefile
        sys/arch/i386/i386: machdep.c sys_machdep.c
        sys/arch/i386/include: pcb.h sysarch.h
Removed files:
        lib/libarch/i386: i386_get_ioperm.2 i386_get_ioperm.c
                          i386_set_ioperm.c
Log message:
Delete i386_{get,set}_ioperm(2) APIs and underlying sysarch(2) bits.
They're no longer used by anything and should let us simplify the TSS
handling.
ok mikeb@ naddy@
```

New system

- I’ve bought Denkovi SNMP controller with 24 I/O (<60 euro)

- IR ans sirens are connected via opto coupler (home made)

New system

- Alarm program has been rewritten in Python. It loops around the SNMP get to detect PIR events. Generate SNMP set to trigger sirens.
- SMS-tools and ftpd and httpd are still there.
- To avoid complicated fsck, this new system is now diskless for the system (webcam pictures are still on a disk).
  - Boot from USB drive
  - Put most file systems in RO
  - Other are in memeory (`mfs`)

How to build a read only OpenBSD ?

- Boot bsd.rd and install a normal OpenBSD system on your connected USB drive
- Adapt the filesystems so that you have 1 filesystems: “/”.
- Reboot and adapt fstab like this:

```
53c5718cfea7b5b4.a / ffs ro,wxallowed 1 1
swap    /var mfs rw,-P=/cfg/var,-s=800m 0 0
swap    /dev mfs rw,-P=/cfg/dev,-s=32m 0 0
swap    /tmp mfs rw,-s=64m 0 0
```

- mkdir /cfg
- cp -Rp /var /cfg/
- cp -Rp /dev /cfg/; cd /cfg/dev ; MAKEDEV all

Read only OpenBSD

- Simple, no ?
- Point of attentions:
  - Since 1GB will be filesystem, please use at least 4GB RAM.
  - Have an USB-2 port. I cannot boot from USB connected on USB-3 (maybe my fault).
  - Before each change, perform "`mount -uw /`" and after "`mount -ur /`"
  - After each pkg_add (or httpd, ...), update your /cfg/var. For example do "`rsync -a –delete /var /cfg`"
- For the rest, the machine is exactly like a normal OpenBSD

Future idea

- Put most of elements under POE (for camera)

Conclusions

- OpenBSD is by far the simplest operating system I’ve worked with since 1998.
- OpenBSD is very flexible: old and new hardware, read-only filesystem, ...
- Man pages are complete, up-to-date and really useful.
- No need to review your whole setup after an upgrade (2x per year).
- Upgrade is FTR and takes +- 30 minutes on average per machine. Delay is mainly because of my network connection.
- Lot of developers are maintaining it and keep it secure (syspatch)
- Do not listen to people saying “bsd is dying, bsd is no more an option today, bsd is just for network tasks, ...”. Make your own evaluation by your self.
- BSD is perfect system for an House security system.

Questions ?

Email: vincent.delft@easytransitions-ict.be

Blog: http://vincentdelft.be

Company: http://easytransitions-ict.be

ET-ICT | Easy Transitions in ICT