CENTRE DE RECHERCHE PUBLIC HENRI TUDOR
www.tudor.lu

VoIP Tutorial - Asterisk

Patrick Harpes

Johannes Hermen

LinuxDays 2006 - 26th January

Agenda

- What is Asterisk?
  - Architecture
- Installing Asterisk
  - Pure VoIP solution
  - Connecting to the outside world
  - Integrating existing ISDN HW
- Starting and managing Asterisk
- SIP Clients
- Dial Plan
  - Syntax
  - Contexts and Extensions
  - Pattern matching
  - Priorities
  - Variables
  - Important Applications
  - Call Parking

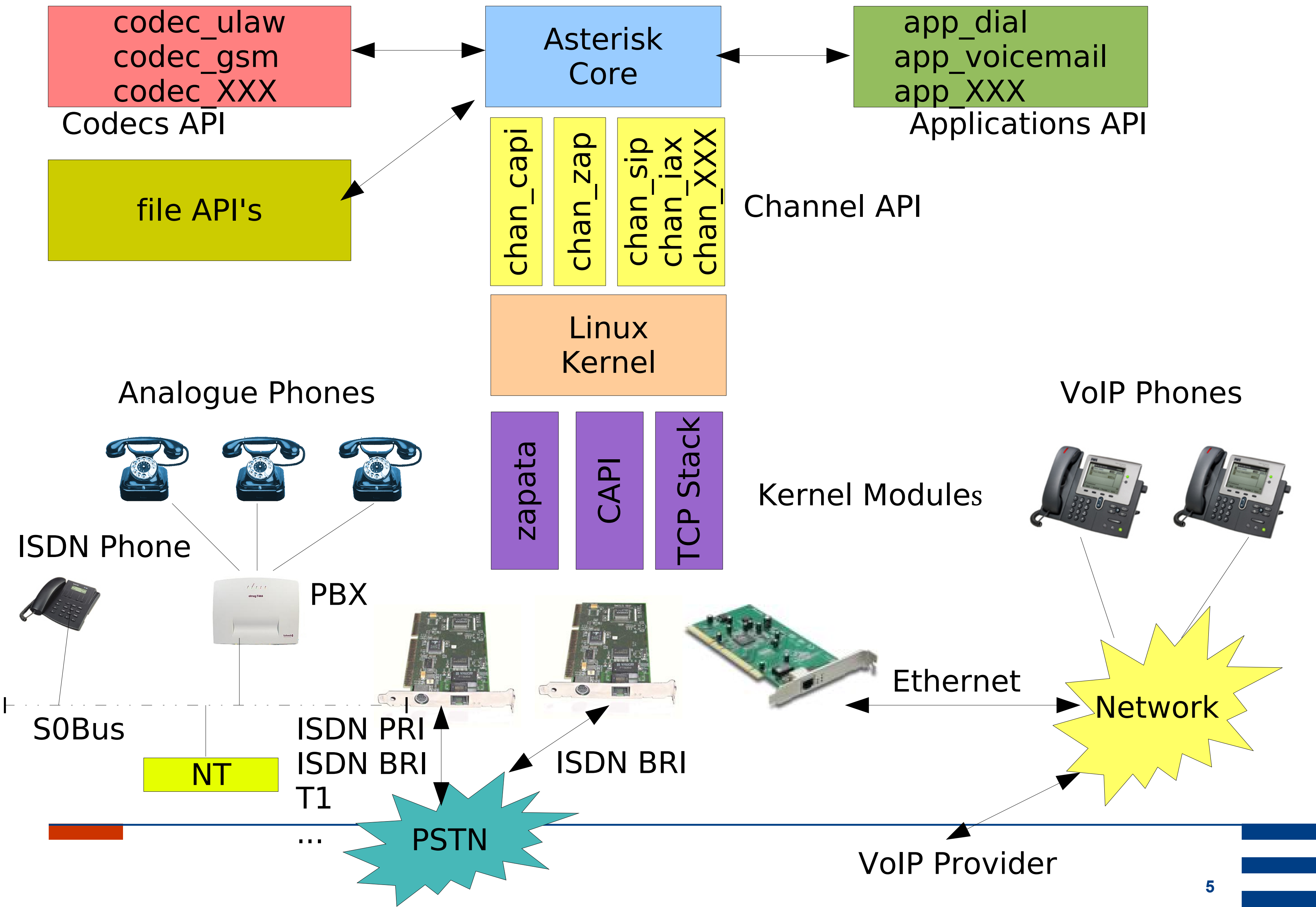
Agenda (2)

- Voice-Mail
- Connecting the outside world
- Connecting using IAX2 Protocol
- Integrating existing ISDN Equipment
- Softphones overview
- Setting up menus

What is Asterisk

- Open Source PBX created by Digium
- GPL License
- Supported OS's
  - Linux
  - BSD
  - Mac OS
  - Solaris
- Features
  - PBX Switching
  - Application Launcher
  - Codec Translator
  - Scheduler and I/O Manager

Asterisk Architecture



Architecture

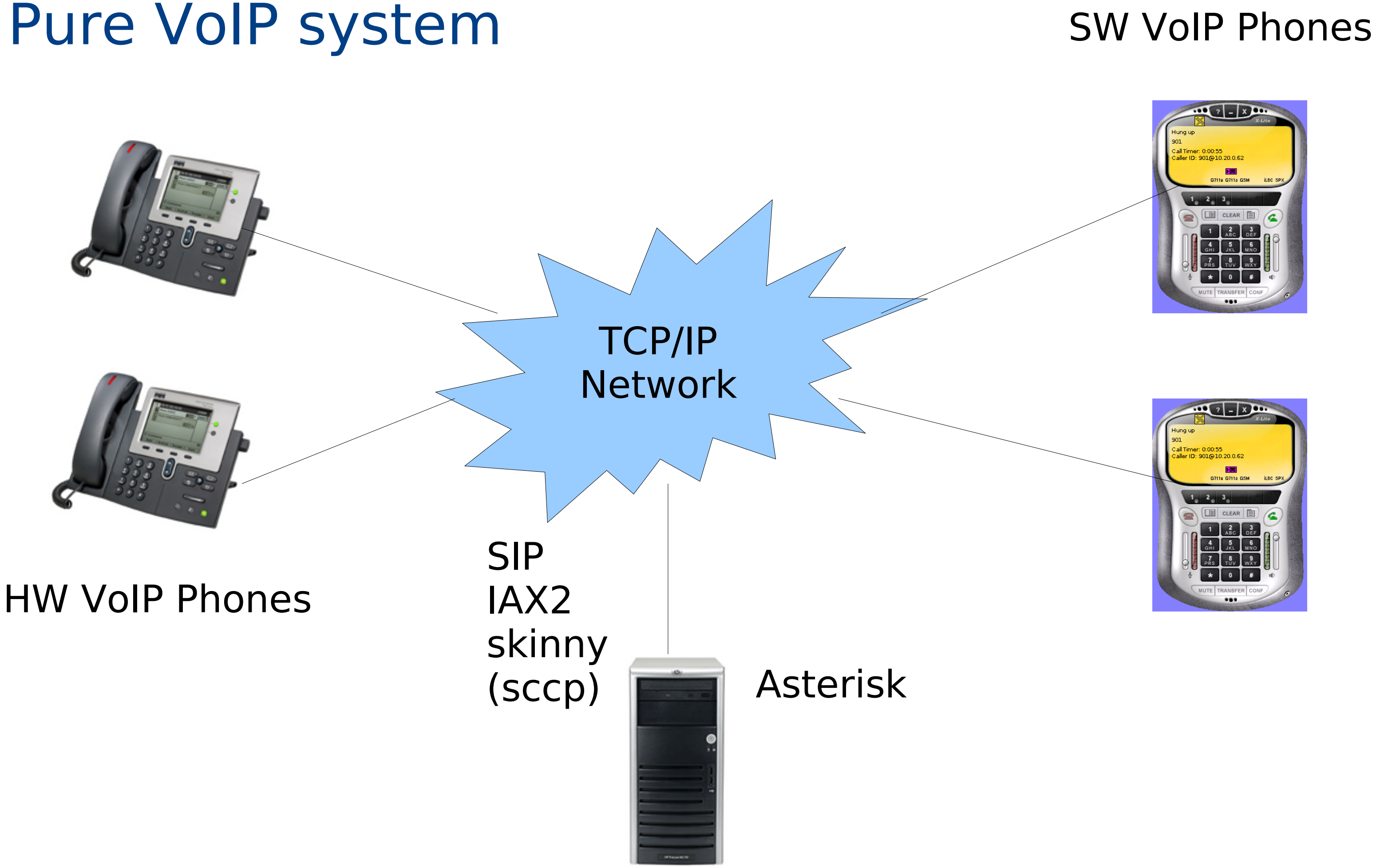
- Four API's defined for loadable modules
  - Channel API
    - handles the type of connection a caller is arriving on
    - chan_sip : supports VoIP SIP
    - chan_iax: supports VoIP Asterisk protocol
    - chan_zapata : supports different Digium ISDN Hardware
    - chan_capi : supports CAPI compliant ISDN cards (Fritz cards)
    - chan_sccp : supports VoIP (Skinny Client Control Protocol) (Cisco)
    - ...
  - Application API
    - each Asterisk application is implemented as module
    - applications are used in the dial plan (see later)
    - app_dial: to dial a number
    - app_voicemail: the voice mail application
    - app_playback: to play music on hold (MOH)
    - ...

Architecture

- Asterisk API's
  - Codec Translators
    - to support various audio encoding and decoding formats
    - codec_gsm: supports GSM format
    - codec_ulaw: suports ulaw format
    - ...
  - File Format
    - handles reading and writing of various file formats

Installing Asterisk

- Different scenarios
- Pure VoIP system



Installing Asterisk

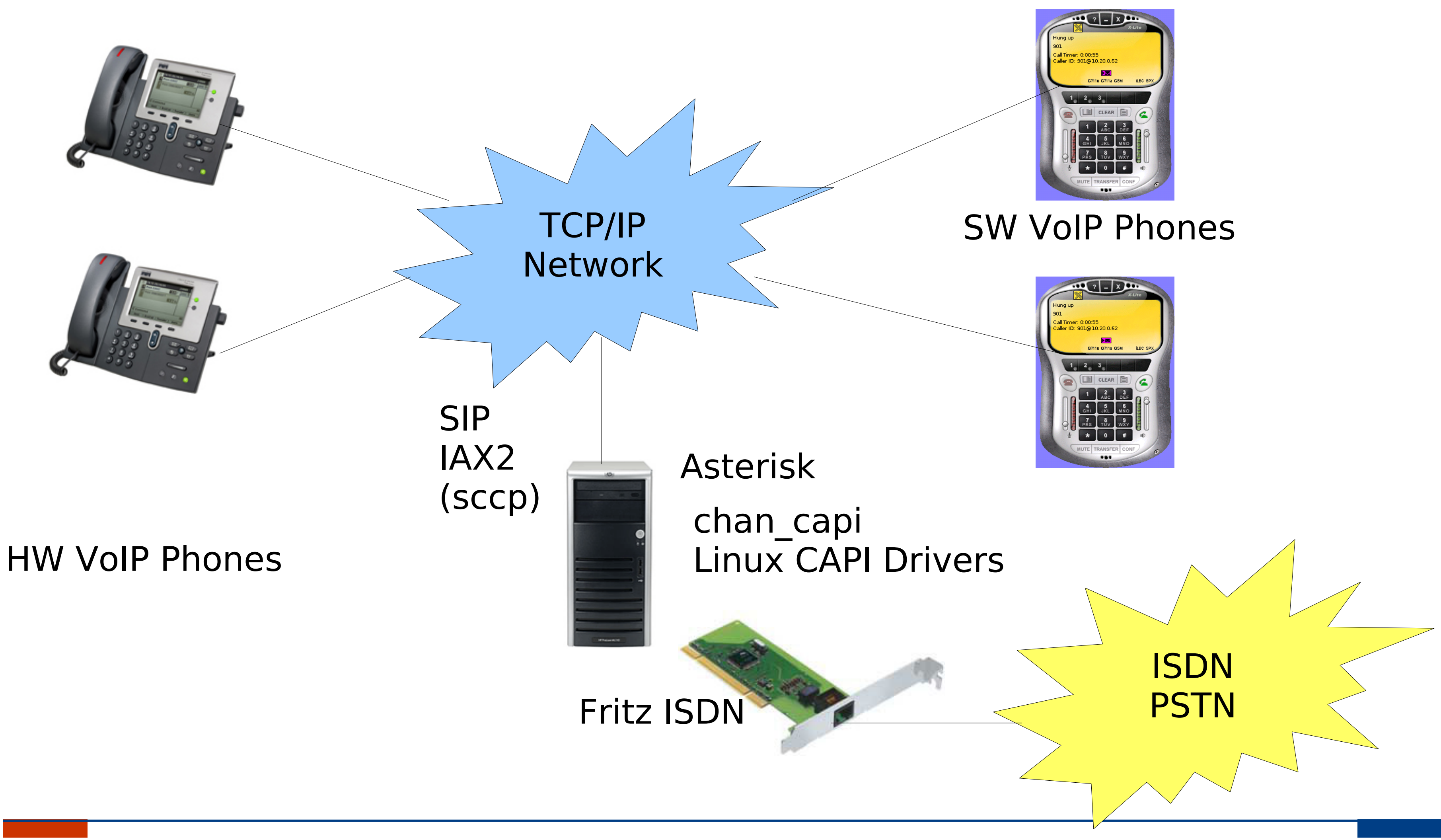
- Pure VoIP Server
  - Files needed
    - asterisk-1.2.X.tar.gz
    - asterisk-addons-1.2.X.tar.gz
    - asterisk-sounds-1.2.X.tar.gz
  - Additional Debian packages
    - libssl developer
    - zlib developer
    - linbcapi20 developer (only for chan_capi)
  - Installation
    - make
    - make install
    - make examples
- This installation is not very useful!

Installing Asterisk

- Scenario 2, connecting Asterisk to the PSTN network using ISDN technology
- Different possibilities
  - cheap solution
    - ISDN PCI FritzCard (2 Channels)
  - Expensive solution
    - Digium Hardware (PRI T1 etc) 30 and more channels
- During this tutorial we only consider the cheap solution
- Why?
  - No ISDN PRI available to test
  - No hardware to test
  - Everyone can test it at home

Installing Asterisk

➤ Asterisk connected via Fritz PCI ISDN Card



Installing Asterisk

- Asterisk with Fritz Card
  - Files needed
    - same files as the pure VoIP Asterisk system
    - chan_capi-cm-0.6.1.tar.gz from sourceforge.net
    - Linux CAPI driver for the card

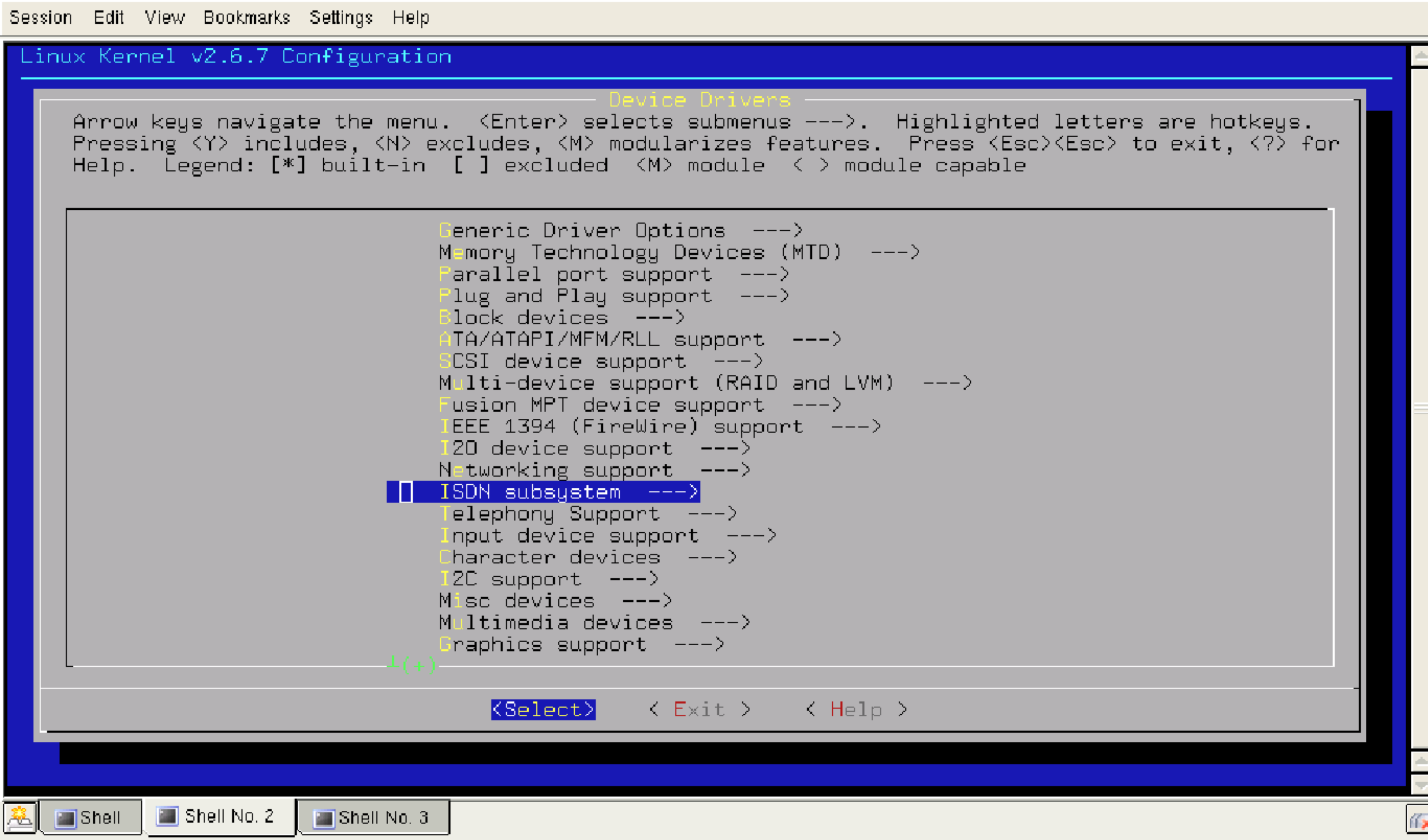
      ftp://ftp.avm.de/cardware/fritzcrd.pci/linux/suse.93/fcpci-suse93.tar.gz
  - Linux Kernel issues
    - If possible use kernel 2.6.X !!
    - You have to compile CAPI support into your kernel
    - Download your kernel source (www.kernel.org)
    - Unpack it into /usr/src/linux
    - run make menuconfig
    - Check the different options to enable CAPI support:

CENTRE DE RECHERCHE PUBLIC HENRI TUDOR www.tudor.lu

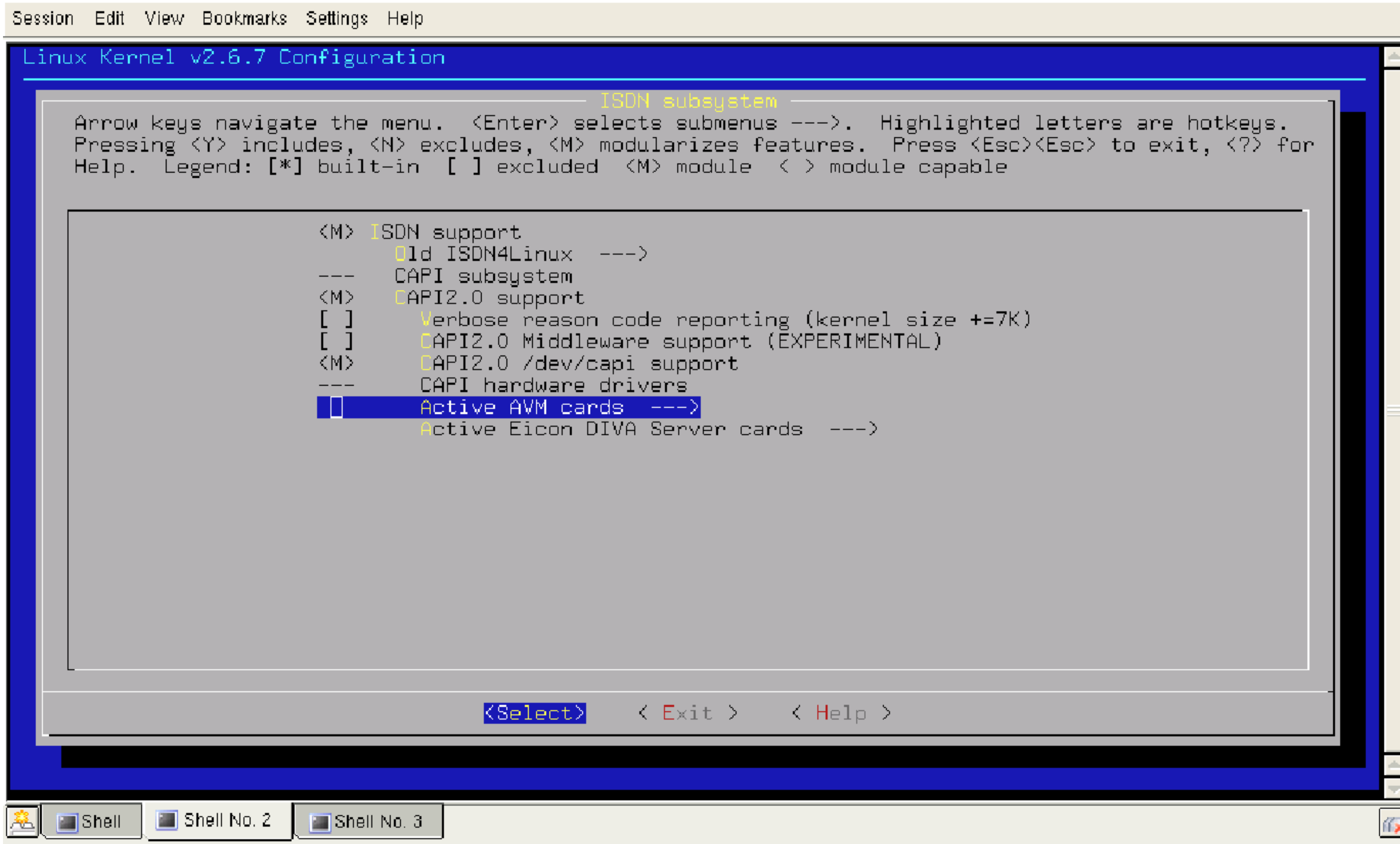
Installing Asterisk

```
Session  Edit  View  Bookmarks  Settings  Help

Linux Kernel v2.6.7 Configuration

                        Code maturity level options
  Arrow keys navigate the menu.  <Enter> selects submenus --->.  Highlighted letters are hotkeys.
  Pressing <Y> includes, <N> excludes, <M> modularizes features.  Press <Esc><Esc> to exit, <?> for
  Help.  Legend: [*] built-in  [ ] excluded  <M> module  < > module capable

            [*] Prompt for development and/or incomplete code/drivers
            [*]   Select only drivers expected to compile cleanly
            [*]   Select only drivers that don't need compile-time external firmware

                      <Select>    < Exit >    < Help >

Shell   Shell No. 2   Shell No. 3
```

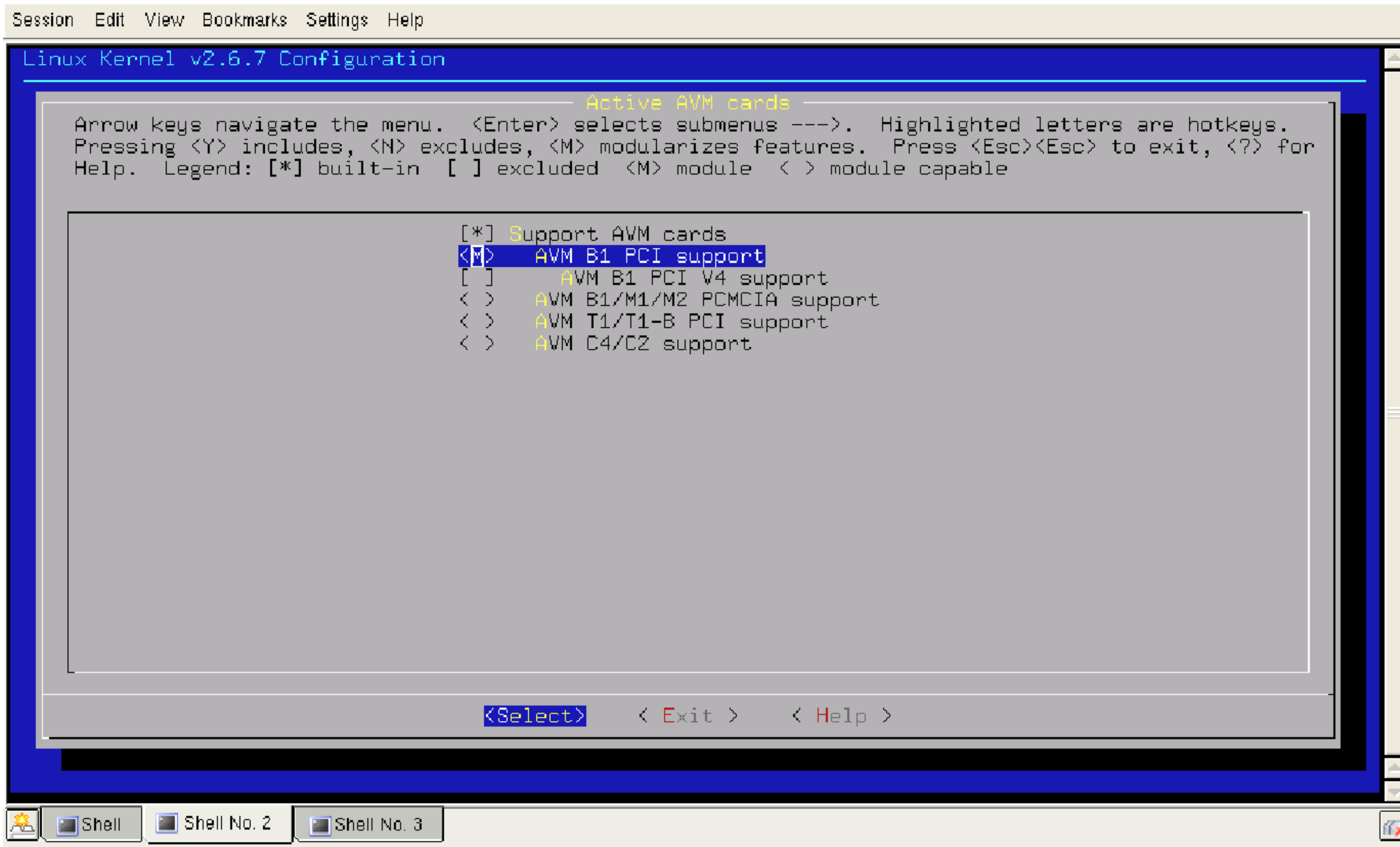
Installing Asterisk



Installing Asterisk



Installing Asterisk



Installing Asterisk

- After the kernel configuration run
  - make bzImage
  - make modules_install
  - copy the kernel image to /boot
  - update your bootloader
  - reboot the system
  - with capiinfo you can test if CAPI is working
- Install the Fritz CAPI drivers
  - Unpack the fcpci-suse93.tar.gz to /usr/src/fritz
  - Compile it using make
  - Install the modules using make install
    - This copies the modules into /lib/modules/2.6.X/extra
  - modprobe fcpci loads the CAPI driver

Installing Asterisk

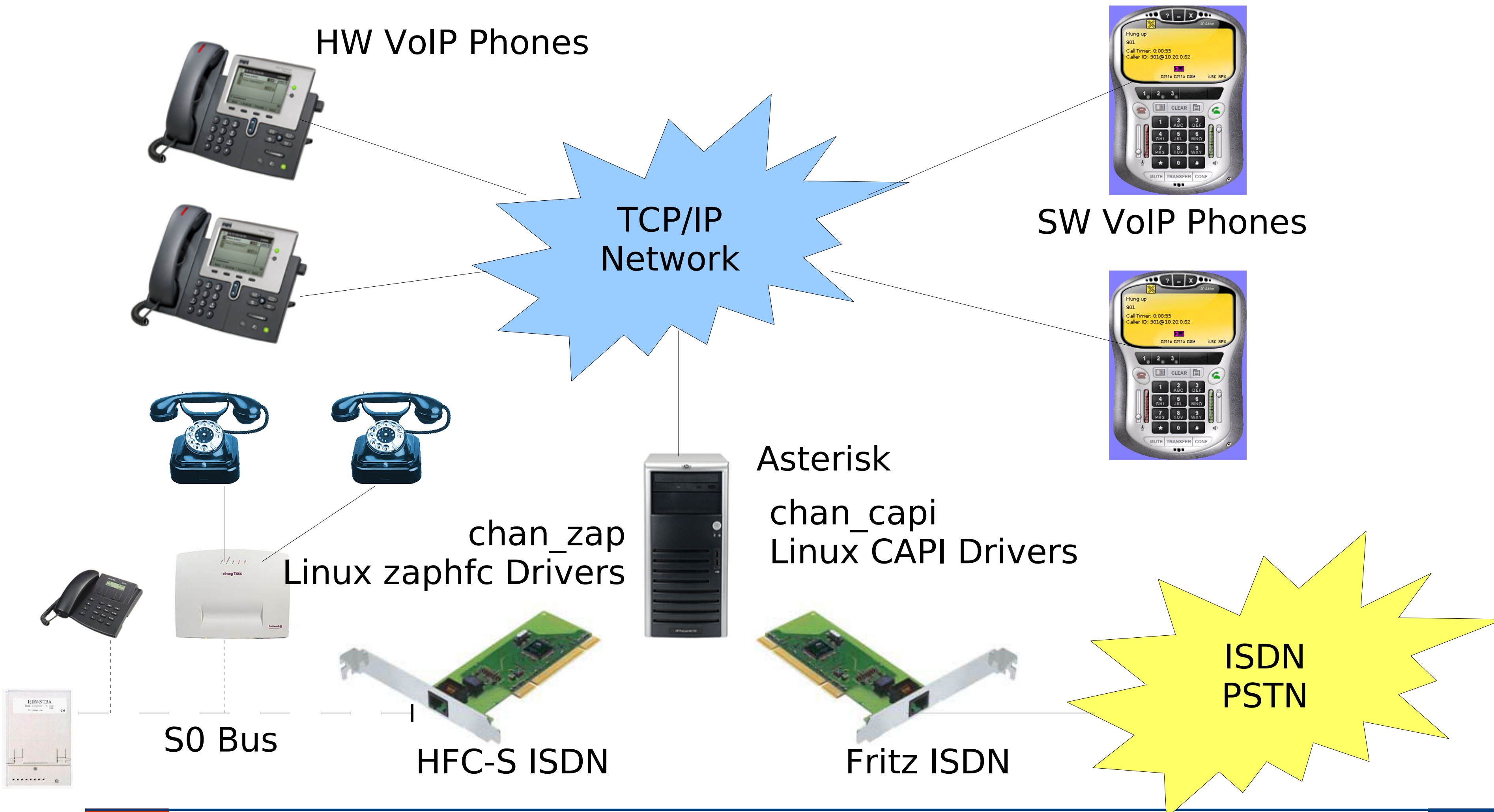
- Install the Asterisk chan_capi module
  - Download the package from sourceforge.net
  - Unpack the package into /usr/local/src
  - Edit the top-level Makefile and make sure INSTALL_PREFIX and ASTERISK_HEADER_DIR point to the correct directory
    - INSTALL_PREFIX=/usr/local
    - ASTERISK_HEADER_DIR=/usr/local/src/asterisk-1.2.X/include
  - run make to compile the package
  - run make install to install the module
  - optional: run make config to have a sample configuration
  - edit Asterisk's module.conf file to add the driver
    - load => chan_capi.so
    - set chan_capi.so=yes in the [global] section of the file

Installing Asterisk

- Kernel 2.4 versus 2.6
  - Timing problem
  - 2.4 needs USB2.0 hardware to produce exact timing
  - 2.6 has timing capabilities in software
- zaptel package
  - Digium hardware support
  - ztdummy driver
    - Applications that need exact timing (Ex. MeetMe())
  - Unpack zaptel-1.2.X.tar.gz into /usr/local/src
  - run make
  - run make install (copies the modules to /lib/modules/2.6.X/extra)
  - modprobe ztdummy

Installing Asterisk

➤ Asterisk integrating existing ISDN hardware



Installing Asterisk

- Installing the asterisk driver for HFC-S ISDN cards
  - Files needed
    - same files as the pure VoIP Asterisk system
    - zaptel-1.2.X.tar.gz zaptel drivers for asterisk
    - bristuff zaphfc drivers and patches for asterisk and zaptel drivers

      http://www.junghanns.net
  - Installing the zaphfc driver
    - patch the asterisk server and the zaptel driver with the patch files from the bristuff package.
    - compile and install the zaphhc kernel module using make
    - load and initialize the zaphfc kernel module

      modprobe zaphfc modes=1

      ztcfg -v
    - test the configuration of the HFC-S ISDN card

      cat /proc/zaptel/1

Starting and managing Asterisk

- Load the Linux modules
  - On Debian add the modules to /etc/modules
    - capi
    - fcpci
    - zaptel
    - ztdummy
- Asterisk can be started in several ways
  - /usr/local/usr/sbin/asterisk (in foreground)
  - /usr/local/usr/sbin/safe_asterisk (in background)
  - Out of the inittab file:
    - as:2:respawn:/usr/local/usr/sbin/asterisk -f
    - restarts automatically after a crash
    - not recommended during setup or test

Managing Asterisk

- asterisk -r connects to the Asterisk daemon
- asterisk -rvvv connects to the Asterisk daemon and make Asterisk more verbose

```
Asterisk 1.2.0, Copyright (C) 1999 - 2005 Digium.
Written by Mark Spencer <markster@digium.com>
=========================================
Connected to Asterisk 1.2.0 currently running on asterisk (pid = 28183)
Verbosity is at least 4
asterisk*CLI>
```

- help to have a list of all commands
- reload to reload the configuration
- stop gracefully to stop the asterisk process
- exit to exit the command line interface

Handling SIP Clients

- Using the SIP channel module, Asterisk is able to act as :
  - a SIP Client: This means that Asterisk registers as a client to another SIP server and receives and places calls to this server.
  - a SIP Server: Asterisk can be configured so, that SIP clients register to the Asterisk server and set up SIP sessions with the server.
  - a SIP Gateway: Asterisk acts as Media gateway between SIP, IAX, H.323 and PSTN connections
- Configuring SIP
  - sip.conf configuration file
  - Each SIP client or server is identified by a block of text that looks like:

Handling SIP Clients

- sip.conf file

```
[xxx]
type=yyy
parameter1 = value
parameter2 = value
```

- [xxx] is the name associated with the SIP client
- Type is either "user", "peer" or "friend".
  - user : is used to authenticate incoming calls
  - peer : is used to authenticate outgoing calls
  - friend : is used for both

Handling SIP Clients

- Example

```
[general]
srvlookup=yes
disallow = all
allow = alaw
allow = ulaw
allow = gsm

[200]
type = friend
secret = mypassword
qualify = yes
nat = no
host = dynamic
canreinvite = no
context = internal
mailbox = 200
```

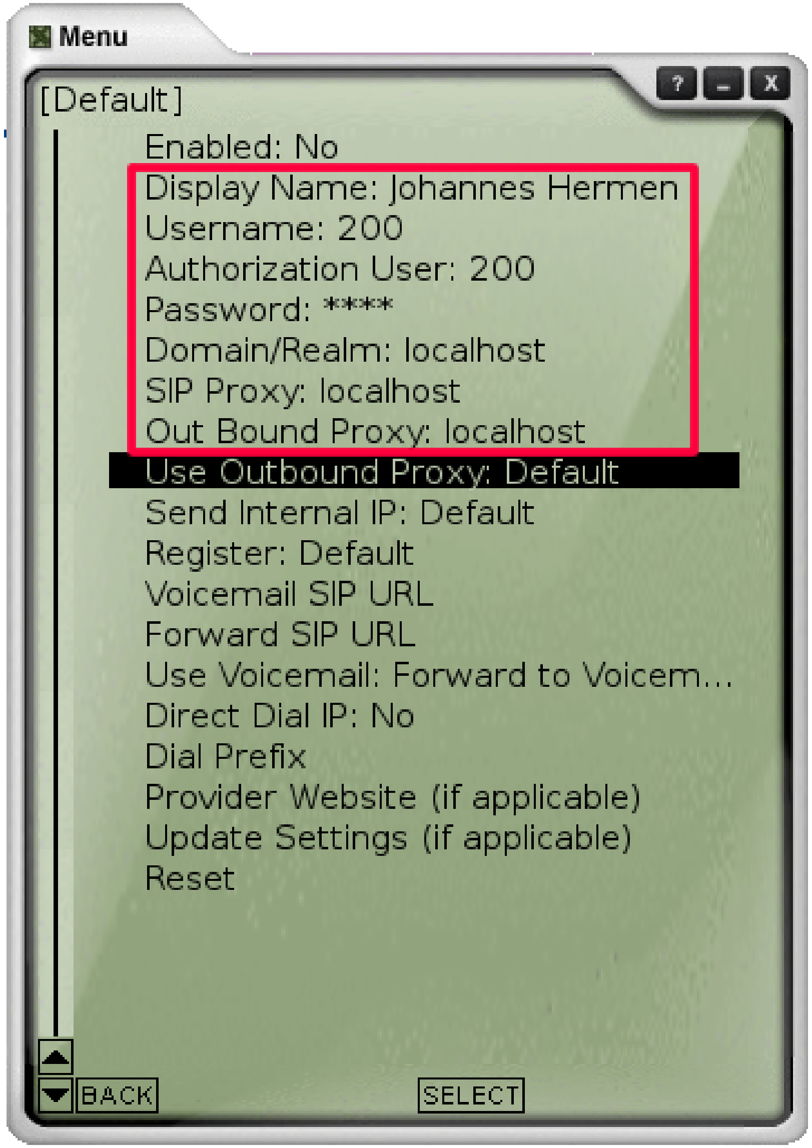
- srvlookup
  - tells Asterisk to make DNS lookup on outgoing calls. (Service Records)
- secret
  - specifies the password for this client
- qualify
  - specifies if Asterisk should send SIP OPTIONS command to check if the device is still online
- nat
  - changes the behaviour of Asterisk for clients behind a firewall. Does not solve the problem if Asterisk is behind the firewall and the client on the outside.
- host
  - the IP address of the client, or 'dynamic' if the client uses dynamic IP addresses
- canreinvite
  - specifies if the media streams passes directly through Asterisk or not
- context
  - the context into an incoming call enters
- mailbox : The extension for the voicebox

Exercise

- Define now one or more SIP clients that use your Asterisk system. All clients should be able to initiate and accept phone calls. They should all be in a context called "InternalSIP". The extensions should be 3 digits long.
- Consider the extensions given to you
- Edit the SIP.conf file
- Connect Asterisk using asterisk -r command
- Use `sip show peers` to check your config
- Reload the Asterisk configuration using internal Asterisk reload command

Configuring the X-Lite Softphone

- Start „xlite“ from the console.
- Follow the setup wizzard.
- Select default speakers/mic.
- Test / adjust volumes.
- Choose LAN connection.
- Finish the wizzard.
- Open the settings dialog.
- Edit the marked settings.
- Close the settings Dialog
- Dial your own SIP number to test the Phone



Dial plan

- extensions.conf file contains the dial plan of Asterisk
- Dial plan is the main configuration file of Asterisk
- It controls incoming and outgoing calls as well as the launch of applications
- Dial plan syntax
  - At the top of the extensions.conf file, general settings could be configured
  - The next section, the globals section defines global variables and their initial values
  - Contexts and Extensions
    - This part of the files defines the dial plan itself
    - The dial plan consists of collections of contexts
    - Each context consists of a collection of extensions

Dial plan

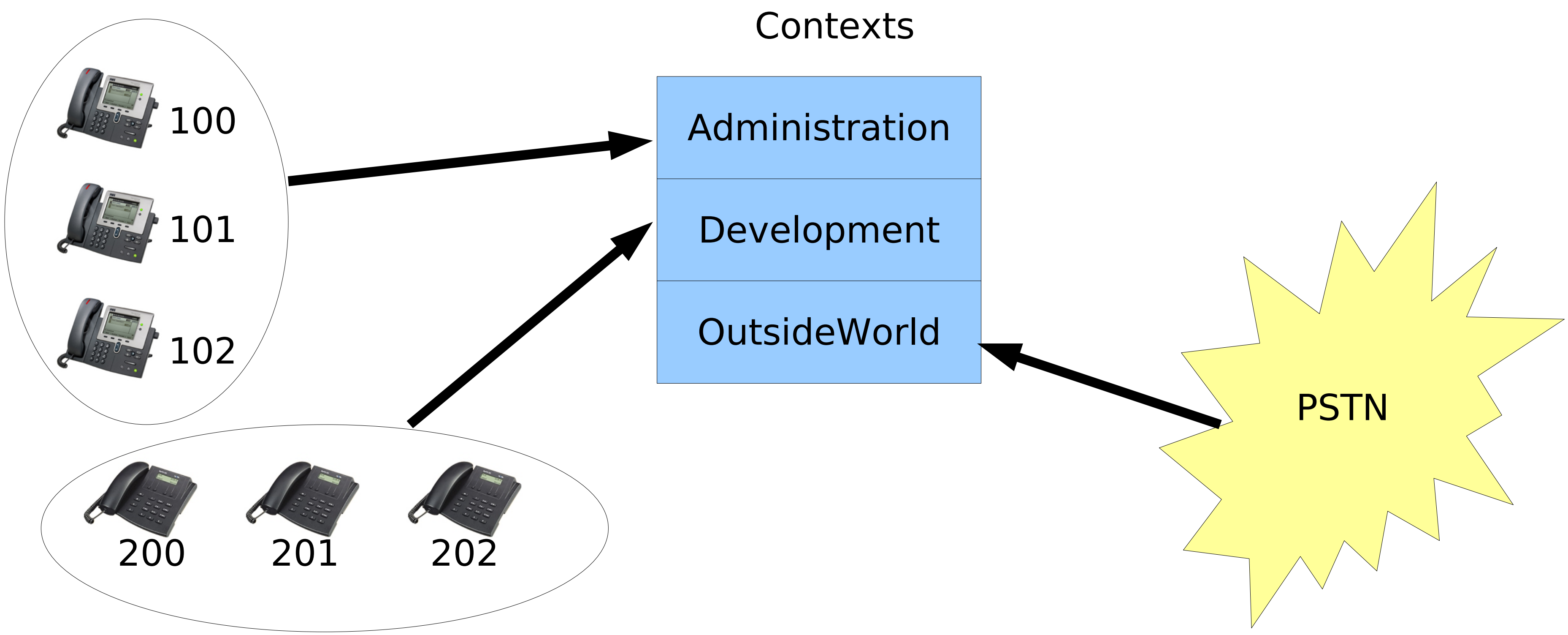
- Simple dial plan example

```
[general]

[globals]

[Internal]
exten => 100,1,Answer
exten => 100,2,Dial(SIP/100,45)
exten => 100,3, Hangup
```

- [general] and [globals] section are empty in this small example
- [Internal] context contains one rule.
  - when someone dials the extension 100 from the context [Internal], the first action is to answer the call.
  - The second action is to dial to the requested extension using the Dial() command
  - After the call has been finished, the call should be terminated using the hangup() command

Dial plan contexts and extensions

- Dial plan consists of collection of contexts
- Contexts definitions are the most important part of the extension.conf file
- A context is just a collection of extensions



Dial plan pattern matching

- Extensions can be simple numbers like 432
- or can be alphanumeric like 'john' or 'B45'
- "normal" phones can only handle numbers
  - In this tutorial an extension will be represented by a number
- Imaging the following situation:
  - You have 500 SIP users, and you have to configure the dialplan:

```
[Internal]
exten => 100,1,Answer
exten => 100,2,Dial(SIP/100,45)
exten => 100,3, Hangup

exten => 101,1,Answer
exten => 101,2,Dial(SIP/101,45)
exten => 101,3, Hangup

...
exten => 599,1,Answer
exten => 599,2,Dial(SIP/599,45)
exten => 599,3, Hangup
```

Dial plan pattern matching

- That's not what you want!!!!
- We have to use pattern matching to solve the problem
- The dial plan supports pattern matching :-))
- Here are the rules:
  - An extension name is a pattern if it starts with '_' (underscore)
  - Special characters for pattern matching
    - X matches any digit from 0-9
    - Z matches any digit from 1-9
    - N matches any digit from 2-9
    - [1237-9] matches any digit in the brackets (in this example 1,2,3,7,8,9)
    - . (point) wild-card, matches one or more characters
    - ! (exclamation) wild-card, matches zero or more characters

Dial plan pattern matching

- Example
  - _61XX Esch Office
  - _63XX Esch Office
  - _62XX Luxembourg Office
  - _7[1-3]XX Wiltz Office
  - _7[04-9]XX Clervaux Office
- Explication
  - All calls starting with 61 or 63 are designated for the Esch office
  - All calls starting with 62 are designated for Luxembourg
  - All calls starting with 71, 72 or 73 are for Wiltz
  - All calls starting with 70, 74, 75, 76, 77 78, 79 are for Clervaux

Dial plan pattern matching

- More example patterns
  - _NXXXXXXX matches a normal 7 digit phone number
  - _9011. matches any string of at least five characters that starts with 9011, but it does not match the four-character string 9011 itself
  - _# matches a single # keypress
  - _. matches everything
  - WARNING: DO NOT USE A PATTERN OF _. AS THIS WILL MATCH EVERYTHING INCLUDING ASTERISK SPECIAL EXTENSIONS LIKE i,t,h etc. Instead use something like _X. or _X which will not match special characters

Dial plan pattern matching

➤ Our example using pattern matching:

```
[Internal]
exten => 100,1,Answer
exten => 100,2,Dial(SIP/100,45)
exten => 100,3, Hangup

exten => 101,1,Answer
exten => 101,2,Dial(SIP/101,45)
exten => 101,3, Hangup

...
exten => 599,1,Answer
exten => 599,2,Dial(SIP/599,45)
exten => 599,3, Hangup
```

```
[Internal]
exten => _XXX,1,Answer
exten => _XXX,2,Dial(SIP/${EXTEN},45)
exten => _XXX,3, Hangup
```

- ${EXTEN} Predefined Channel Variable that contains the current extension

Dial plan priorities

- Priorities are numbered steps in the execution of each command to make an extension.
- Priority numbers starts at 1 and increment consecutively for each line in the context.
- Each priority represents one specific application

```
[Internal]
exten => 555,1,Answer
exten => 555,2,Playback(tt-weasels)
exten => 555,3,Voicemail(44)
exten => 555,4,Hangup
```

  - "exten =>" tells the dial plan that the next thing it sees will be a command
  - 555 are the actual digits received (i.e. what the caller dialled)
  - "1", "2", "3", "4" represent the priority, which determines the order in which commands for that extension will be executed
  - the last parameter of the "exten" line is the command itself

Dial plan variables

- Asterisk can make use of global and channel specific variables
- Variables are referenced in the dial-plan using the syntax: ${foo}
- Variables names ma be any alphanumeric string beginning with a letter
- User defined variable names are not case sensitive
  - ${foo} = ${FOO}
- Astersik defined variables are case sensitive
  - ${exten} != ${EXTEN}
- A list of Asterisk defined variables in the support

${EXTEN} Variable

- ${EXTEN} : the current extension
- ${EXTEN:n}: the current extension excluding the first n digits
  - Example:
  - ${EXTEN} = 012345
  - ${EXTEN:1} = 12345
  - ${EXTEN:2} = 2345

Asterisk Applications

- Applications are the workhorses of the dial plan
- Each application performs a specific action on the current channel.
- Syntax:
  - exten => <extension>, <priority>, <application>
- The parameter list of the application depends on the application

Important Applications

- Answer()
  - If the channel is ringing, answer it, otherwise do nothing
- Hangup()
  - Unconditionally hangs up a given channel
- Playback(filename, options)
  - Plays the specified sound file
  - Sound files are stored in /var/lib/asterisk/sounds
  - Example
    - exten => 500,1,Playback(hello-world)

Exercise

- Create a dial plan that does the following
  - All incoming calls to extensions 800 - 849 should start the playback of tt-weasels
  - All incoming calls to extensions 850 - 899 should start the playback of hello-world
  - Use pattern matching!
  - After the playback is finished, you should hangup the call
  - Using your SIP phone, test the configuration

Dial() command

- At temps to establish a new outgoing connection on a channel, and then link it to the existing input channel
- Dial(type/identifier, timeout, options, URL)
  - type specifies the channel to use (CAPI, Zap, SIP, ...)
  - identifier specifies the phone number
  - timeout parameter is optional. It specifies a maximum of time in seconds, that the Dial() command is to wait for a channel to answer. If not specified Dial() waits indeinitely.
  - options parameter is optional -> see Call Parking
  - URL parameter is optional. Can be used to send a additional URL to the called party
- Example

```
exten => 100,1,Answer()
exten => 100,2,Dial(SIP/100,45)
exten => 100,3,Hangup()
```

Exercise

- Create a dial plan to dispatch incoming calls
  - Respect the extensions that have been distributed
  - 1st Step: create a static entry for one of your extensions
  - 2nd Step: create a generic entry to handle your whole range
- Test your configuration by calling yourself using your soft phone

Call parking

- Receptionist feature
- All entering calls routed to the receptionist
- The receptionist answers the call and dispatches it manually to the asked extension
- During dispatching, the caller has to wait and MOH is played
- How to configure:
  - edit the features.conf file and make sure the following parameters are in:

```
[general]
parkext => 700                  ; What ext. to dial to park
parkpos =>701 - 720             ; ext. to park calls on
context => parkedcalls          ; the context paked calls are in
parkingtime => 45               ; max sec. a call can be parkedfor
```

Call parking

- edit the extensions.conf and include the parkedcalls file:

```
[internal]
include => parkedcalls
```

- Add the 't' and 'r' flag to the corresponding Dial() commands
  - 't' means that the called user can transfer the call
  - 'r' tells the calling party that the extension is ringing

```
[internal]
include => parkedcalls

exten => _XXX,1, Answer()
exten => _XXX,2,Dial(SIP/1,40,tr)
exten => _XXX,3,Hangup()
```

Voice mail

- Voice mail feature allows to create voice mail boxes for each user.
- 1st Step to configure the voice mail box is to set-up voicemail.conf file
- Example

```
[general]
format = gsm|wav
attach=yes
maxgreet=30
maxmessage=90

[default]
100 => 1111, Patrick, harpes@linuxdays.lu
200 => 1234, Johannes, johannes@linuxdays.lu
```

Voice mail

- [general] section
  - has to be in the voicemail.conf file
  - common configuration for all users
  - attach: enables Asterisk to send an email containing the audio message to the user
  - format: defines the format in which the audio message will be stored. For email messages the first specified format will be used
  - maxgreet: maximum length in seconds of the greeting message
  - maxmessage: maximum length in seconds of the message the caller leaves

Voice Mail

- [default] section
  - defines the mail boxes for the users
  - format:
    - voicebox_number => password, user_name, email_address
- extensions.conf
  - specify that a call should be forwarded to a voice box

```
exten => _XXX,1,Answer()
exten => _XXX,2,Dial(SIP/${EXTEN},45)
exten => _XXX,3,Voicemail(u${EXTEN})
exten => _XXX,4,Hangup()
```

Voicemail() command

- Records the channel, saving an audio file in a given voice-mail box number.
- The voice-mail box number may be preceded by one or more flags:
  - 's' flag: causes the instructions to be skipped
  - 'u' flag: causes the unavailable message to be played
  - 'b' flag: causes the busy message to be played
- The messages will be stored in
  - /var/spool/asterisk/voicemail/context/boxnumber/INBOX
  - Control disk space!!!

Accessing the voice-mail box

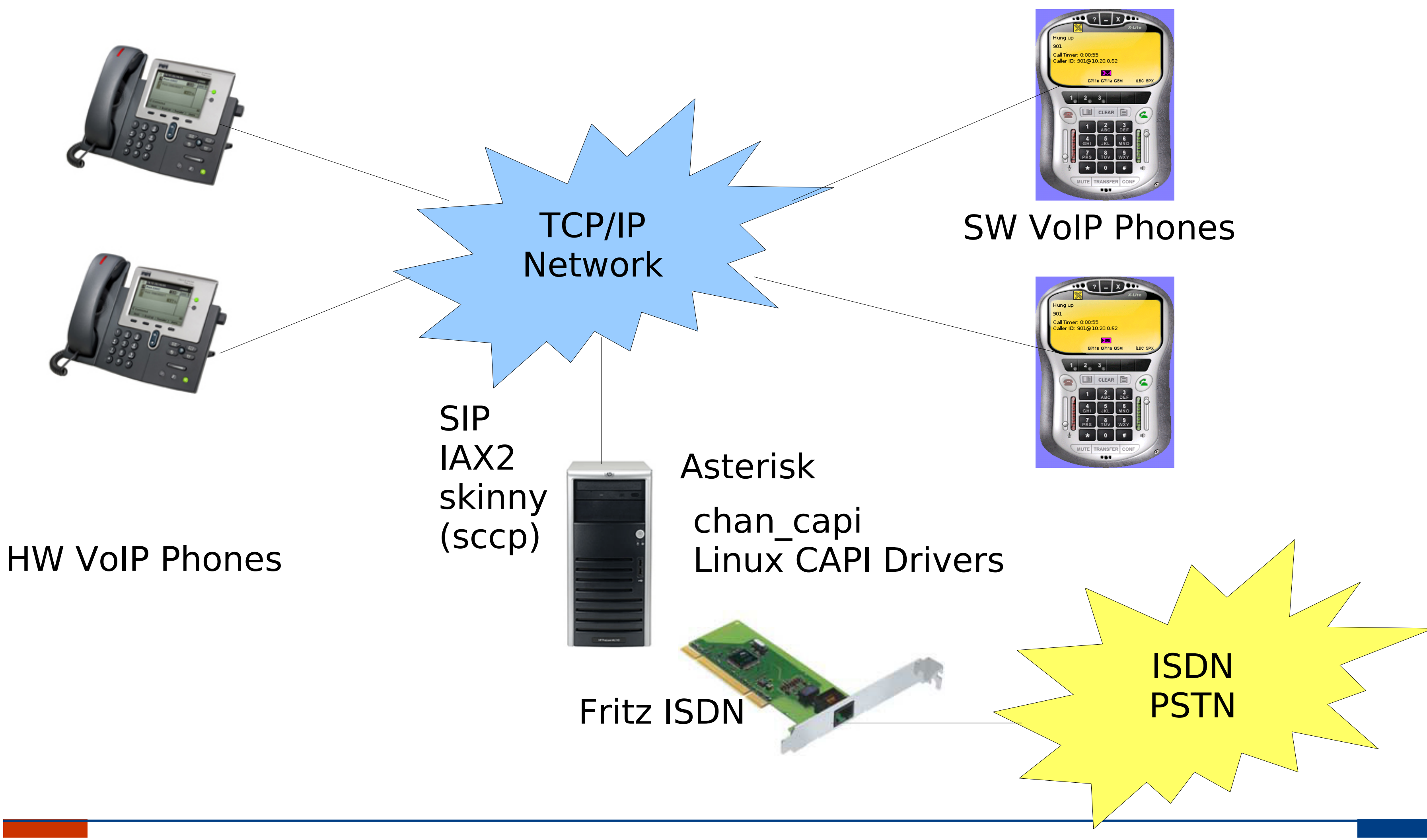
- Where to configure Asterisk to let the user access his voice-mail box?
- extensions.conf of course
- VoiceMailMain() application handles this
  - VoiceMailMain([s]mailbox@context)
  - if the 's' option is present, the password check is skipped

```
; Indirect access to the voice-mail box
exten => 5000,1,Answer()
exten => 5000,2,VoiceMailMain()
exten => 5000,3,Hangup()

; Direct access to the voice-mail box
exten => _5XXX,1,Answer()
exten => _5XXX,2,VoiceMailMain(${EXTEN:1})
exten => _5XXX,3,Hangup()
```

Exercise

- Modify your dial plan in manner to switch to the voice-mail if the user is unavailable
- Extend the dial plan to let the user access his voice-mail box
- Test it using your soft phone

Connecting to the outside world



CAPI Configuration

- capi.conf

```
[general]
nationalprefix=00
internationalprefix=000
rxgain = 0.8
txgain = 0.8

[ISDN1]
isdenmode=msn
msn = 435253
incomingmsn = *
context = capi-in
softdtmf = 0
controller = 1
devices = 2
echocancel = yes
```

- IMPORTANT
  - To activate this configuration you have to restart Asterisk!

extensions.conf

- To let the SIP users phone via ISDN, prefix=0

```
[Internal]
exten => _0.,1,Answer()
exten => _0.,2,Dial(CAPI/contrl1/435253:${EXTEN:1})
exten => _0.,3,Hangup()
```

- Incoming ISDN calls (msn=435253)

```
[capi-in]
exten => 435253,1,Answer()
exten => 435253,2,Wait(1)
exten => 435253,3,Playback(enter-ext-of-person)
; Handle the extension the user enters using DTMF tones
exten => _XXX,1,Dial(SIP/${EXTEN})
```

Connecting to another Asterisk

- IAX Protocol

Configuring IAX protocol

- iax.conf

```
; Luxembourg configuration
[general]
port=5036
disallow=all
allow=ulaw
allow=alaw
allow=gsm
bandwidth=high

[Luxembourg]
type=user ; incoming
secret=1234
context=default
```

```
; Esch configuration
[general]
port=5036
disallow=all
allow=ulaw
allow=alaw
allow=gsm
bandwidth=high

[Esch]
type=user ; incoming
secret=4321
context=default
```

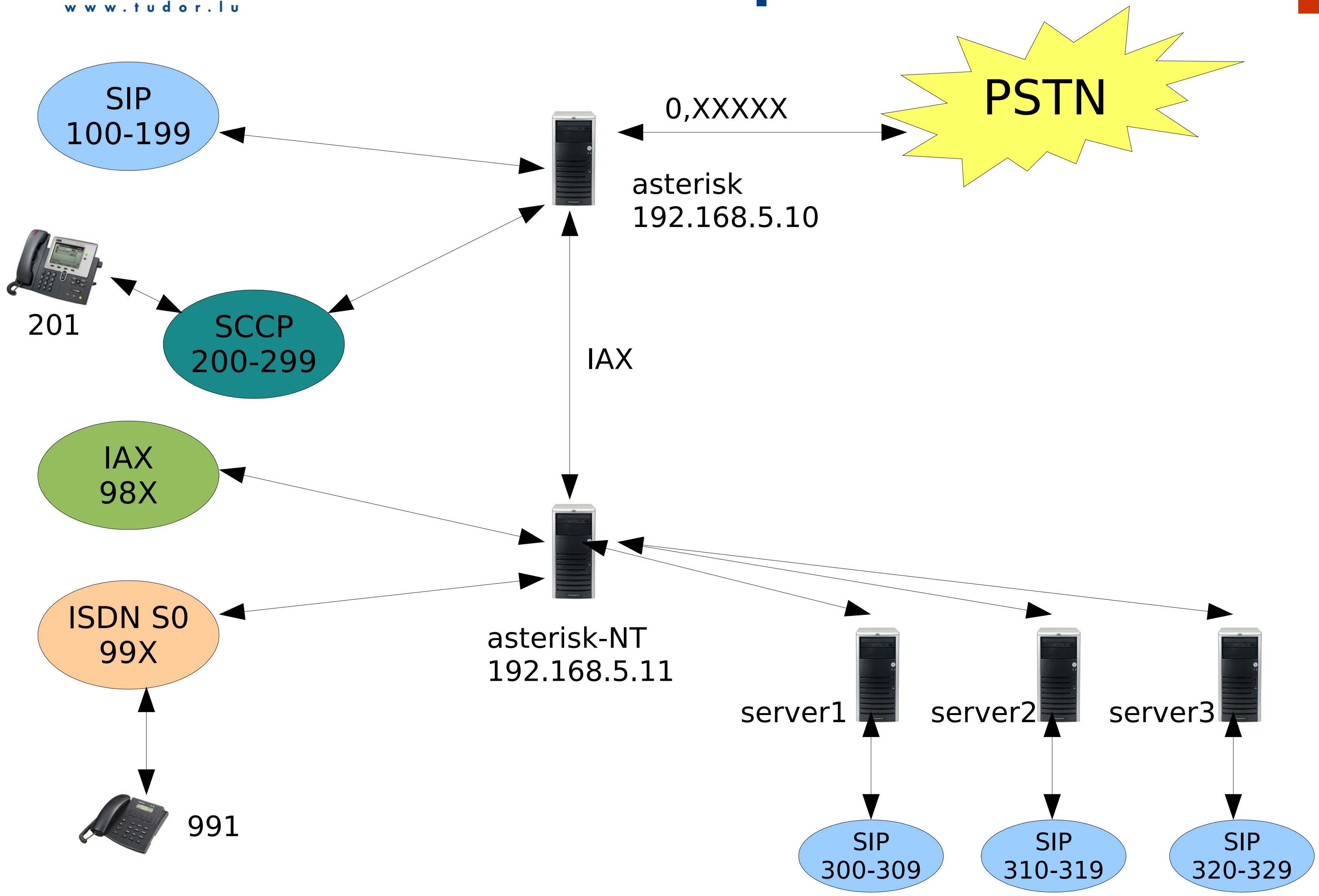
Configuring IAX protocol

➤ extensions.conf

```
; Luxembourg  SIP users 100 -199

[default]
exten => _1XX,1,Answer()
exten => _1XX,2,Dial(SIP/${EXTEN})
exten => _1XX,3,Hangup()

exten => _2XX,1,Answer()
exten => _2XX,2,Dial(IAX2/Esch:4321@esch-alzette.tudor.lu/${EXTEN})
exten => _3XX,3,Hangup()
```

```
; Esch  SIP users 200 -299

[default]
exten => _2XX,1,Answer()
exten => _2XX,2,Dial(SIP/${EXTEN})
exten => _2XX,3,Hangup()

exten => _1XX,1,Answer()
exten => _1XX,2,Dial(IAX2/Luxembourg:1234@luxembourg.tudor.lu/${EXTEN})
exten => _1XX,3,Hangup()
```

Concrete Example



Exercise

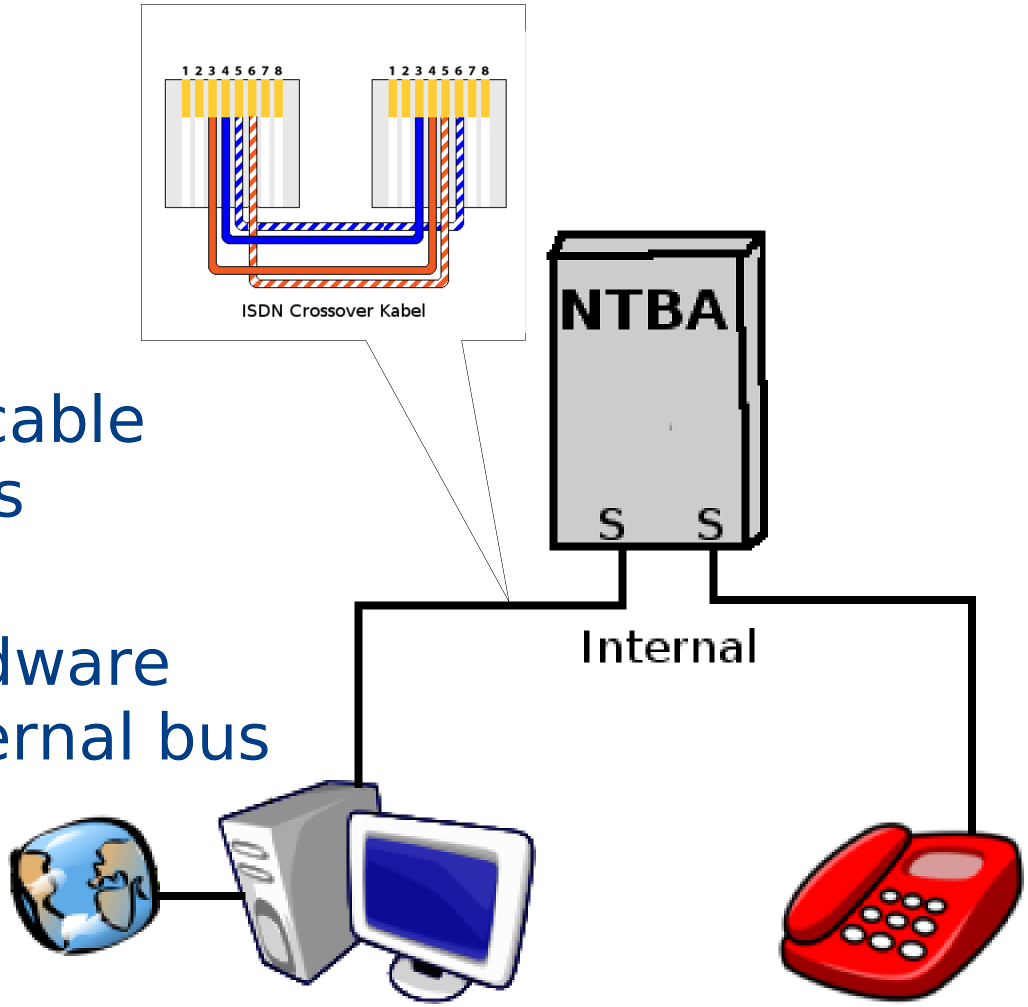
- Configure your system in a manner to integrate it into this system
  - Edit the iax.conf file
  - Edit the extensions.conf file
  - test the configuration
    - Phone to the ISDN phone
    - Phone to the VoIP phone
    - Phone to the outside world
    - Phone to your neighbour

```
[general]
disallow=all
allow=ulaw
allow=gsm
allow=adpcm

; local user acount for connection incoming
; from other asterisk servers

[asterisk-nt]
type=user
secret=1234
context=default
```

Integrating old ISDN equipment

Setup the hardware

- HFC-Card acts as NT
- NTBA powers the bus
- Special X-Over ISDN cable to connect card to bus
- Connect an ISDN hardware phone to our new internal bus



Integrating old ISDN equipment

- Configuring asterisk for ISDN phones
  - editing the zapata.conf to access the internal ISDN bus.

```
[channels]
channel => 1-2
group = 1
switchtype = euroisdn
signalling = bri_net_ptmp
pridialplan = local
immediate = no
overlapdial = yes
echocancel = yes
context = default
```

- Card uses IDSN channels 1-2.
- switchtype and signalling set to fit european phones.
- overlapdial, pridialplan and immediate allow dialling after picking up the phone.
- echocancel reduces echos on line.
- All calls from the ISDN phones are routed to the default context.

Integrating old ISDN equipment

- Configuring asterisk for ISDN phones
  - editing the extensions.conf to route calls from/to the phone

```
; all connections start here
[default]
.....
; 9xx numbers belong to local IDSN phones
exten => _9XX,1,GoTo(isdn-phones,${EXTEN},1)
.....
; Dial local ISDN Phones
[isdn-phones]
exten => _X.,1,Answer()
exten => _X.,2,Dial(Zap/g1/${EXTEN})
exten => _X.,3,Hangup()
```

- All calls start in context default.
- If number starts with 99x goto isdn-phones section.
- Dial the phones on the zap channel with the given number.

SoftPhones - Overview

- **SIP – Softphones**
  - **GnomeMeeting:** http://www.gnomemeeting.org/
    Linux/Gnome – GPL License
  - **Kphone:** http://www.wirlab.net/kphone/
    Linux/KDE – GPL License
  - **SFLphone:**http://www.sflphone.org/
    Linux (Windows, Mac soon) – GPL License
  - **X-Lite:**http://www.globalipphones.com/
    Windows, Linux, Mac OS X, Pocket PC – Freeware
  - **SJPhone:** http://www.sjlabs.com
    Windows, (Linux), Mac OS X, Pocket PC - Freeware
- **IAX – Softphones**
  - **KIAX:** http://kiax.sourceforge.net/
    Linux/KDE – GPL License
  - **Idefisk:**http://www.asteriskguru.com/tools/idefisk_beta.php
    Windows – Free usable for personal/commercial purposes

Setting up menus

- Using DTMF tones on the phone the users could navigate through menus
- Example

```
[default]

...
exten => _87X,Goto(Menu,${EXTEN},1)
...

[Menu]
exten => 870,1,Answer()
exten => 870,2,Playback(press-1)
exten => 870,3,Playback(for)
exten => 870,4,Playback(time)
exten => 870,5,Playback(press-2)
exten => 870,6,Playback(for-the-weather)

exten => 1,1,SayUnixTime()
exten => 1,2 Hangup()

exten => 2,1,Playback(today)
exten => 2,2,Playback(rainfall)
exten => 2,3,Hangup()
```

Exercise

- Create a menu for several services a user can access