**Universal Contact.sh实施记录**

**Process 1**

**题目描述：**

**A背景：**

随着移动互联网与大数据技术的飞速发展，海量数据时刻记录着人们生产生活的轨迹。这些数据在经过分析、挖掘等处理后，可以通过为网民提供个性化、精准性的服务来创造巨额的社会财富。正是由于数据可以通过：集中、流动、共享的基本路径来创造价值，所以数据本身就具有价值。在此过程中，如何规范企业之间的数据共享，如何解决数据控制者与数据主体之间合规性问题，如何保护广大用户的个人信息，防止个人信息被滥用等问题受到社会各界广泛关注。以互联网为代表的新一代信息通信技术与实体经济深度融合，在推动产业提质增效、提升民生服务水平、促进社会治理能力提升的同时，个人信息面临过度采集、违规使用、非法泄露等问题亟待解决。

**B问题：**

一方面，通讯录是连接真实人与虚拟人之间的重要结点，是独立人与其人际关系网络的映射实体，是一个人在社交网络之中最为重要的隐私；另一方面，通讯录也是互连网公司实现数据增殖与财富增值的必由之路，是各大公司的兵家必争之地。用户对于隐私的强烈保护欲望与企业希望通过获取用户通讯录来创造更多财富的愿景相矛盾，使得越来越多的公司高举为用户服务的大旗，行着窃取公民通讯录隐私的苟且之事，构建一个安全方便的通讯录成为当务之急。

**C解决思路：**

我们团队决定运用shell脚本在计算机上实现一个方便实用的通讯录程序。

程序通过add、contact\_add、add\_command和del、contact\_del、del\_command函数，运用了链表数据结构实现了通讯录成员的增加与删除操作。编写算子用于直接操作数据库，同时支持交互模式和命令行模式。

类似地，程序通过edit、contact\_edit和edit\_command函数，运用了链表数据结构实现了对于通讯录成员进行修改的操作。程序通过构建contact数组的方式实现了对于通讯录成员快速查找操作。

考虑到传统通讯录在使用中的缺陷，除了上述常规操作之外，我们还实现了如下特色功能：为了解决疫情期间居家办公因忘记时间而错过线上会议的问题，我们通过alarm函数和crontab命令实现了定时提醒功能。

**成员分工：**

脚本编写：张超凡（功能函数编写）、杨淯而（加解密函数编写）

张思宁（命令行函数编写）、林文豪（基础函数编写）

脚本测试：张超凡

文档撰写：张思宁

PPT制作：林文豪

PPT修改：杨淯而、张思宁、张超凡

文档修改：杨淯而

讲解演示：杨淯而

其他事项：张思宁、林文豪

**本次总结：**

通过这次Universal Contact脚本的编写，我们在写代码、代码分析以及程序调试的过程中，巩固了常用的Linux命令与知识，也接触到了一些新命令和已学命令的新用法，加深了我们对于正则表达式的理解与应用。让我们更加深刻的认识到脚本语言相较于传统底层语言的简易性特点，如在C++中实现复杂的函数在脚本中可以简单的简写为几行代码，并且可以快速部署运行，节省了大量的部署时间。

在测试过程当中，我们发现对通讯录内容进行简单的明文存储存在较大风险，容易被木马程序进行非法拷贝导致个人信息的泄漏，同时也可能因为操作不当通过移动设备发生泄漏与传播，由此看来，对程序增加自动加解密功能十分的有必要。

**Process 2**

**优化更新：**

主要目标：针对Process1中提出的明文存储漏洞进行修复。

目标分析：1）实现对用户数据存储时的加密功能。

2）实现加解密操作对用户透明，保持程序的易用性。

3）加解密操作需要高效迅速，保证程序的快速反应。

**解决思路：**

对目标1采用在脚本中内置加解密函数的方式来保护使用者的个人信息。

对目标2通过在关键操作前后自动调用加解密函数的方式来实现加解密操作对用户透明。

对目标3通过选取函数加密的方式使得程序可以在保证用户数据安全的情况下仍能保持较高的响应速度。对于存储的数据，由于涉及加解密，不能单纯地使用无陷门的单向函数进行加密，因此使用base64进行加解密。同时，为了防止启动程序被人为更改注入恶意代码，我们在启动时检测程序名的base64值，同时将自身某一单向哈希函数值并与云端单向哈希函数值进行比较。选用的函数由脚本随机决定，这属于函数加密的过程。在未来，要查找程序的入侵痕迹或进行取证，只需重新执行函数，得到的值即可判断是否有被更改过。这样既可以保证用户的隐私，又能进行验证，做到了“我不知道里面的内容，但我能验证这个内容是脚本产生的”。

**附录（最终代码）：**

#!/bin/bash

PROGRAMNAME="Universal Contact"

SCRIPTPATH=$(readlink -f $0)

SCRIPTNAME=$(basename "${SCRIPTPATH}")

ARGV=( "$@" )

ARGC=$#

*# Use the absolute path to decide the diectory where the database files store*

DB\_DIR\_PATH="$(dirname "${SCRIPTPATH}")/db/"

*# Global varribles*

USERNAME=""

DB\_FILE\_PATH=""

CONTACT\_LENGTH=0 *# The number of contacts in the database*

CONTACT\_DATE=() *# Contact Date field array*

CONTACT\_ALARM=() *# Contact Alarm field array*

CONTACT\_NAME=() *# Contact name field array*

CONTACT\_GROUP=() *# Contact group field array*

CONTACT\_PRI=() *# Contact priority field array*

CONTACT\_PHONE=() *# Contact phone number field array*

CONTACT\_WECHAT=() *# Contact wechat field array*

CONTACT\_QQ=() *# Contact qq field array*

CONTACT\_EMAIL=() *# Contact email field array*

CONTACT\_COMPANY=() *# Contact company field array*

CONTACT\_NOTE=() *# Contact notes field array*

NUMREGEX='^[0-9]+$'

CRON\_DIR\_PATH="/var/spool/cron/crontabs"

ADD\_HELP="${SCRIPTNAME} add TITLE CONTEXT"

LIST\_HELP="${SCRIPTNAME} list [INDEX]"

EDIT\_HELP="${SCRIPTNAME} edit INDEX TYPE NEWCTX"

DEL\_HELP="${SCRIPTNAME} del INDEX"

ALARM\_HELP="${SCRIPTNAME} alarm MONTH-DAY-HOUR-MINUTE/HOUR-MINUTE"

*## Exit code*

EXIT\_SUCCESS=0

EXIT\_FAILULRE=1

EOF=255

function espeak\_w()

{

 espeak "$1" 1>&2 2>/dev/null

}

function info()

{

 echo -e "\033[32;1mINFO\033[0m: $1"

}

function error()

{

 echo -e "\033[31;1mERROR\033[0m: $1"

}

function print\_title()

{

 [[ $ARGC > 0 ]] && return *# if [[ $ARGC > 0 ]]; then return; fi*

 clear

 echo -n '/'

 printf '\*%.0s' {1..20}

 printf " \033[1;31m${PROGRAMNAME}\033[0;0m "

*#printf " \033[32;5m\033[47m\033[1m${PROGRAMNAME}\033[0m "*

 printf '\*%.0s' {1..20}

 echo '/'

}

function waituser()

{

 [[ $ARGC > 0 ]] && return

 printf '\n\033[5mPlease press any key to continue...\033[0m'

 read -n 1 -s

}

*# Get username*

function get\_username()

{

 USERNAME=$(whoami)

 info "Logged as ${USERNAME}"

}

function get\_cron()

{

 if [[ ! -f $CRON\_DIR\_PATH ]];

 then

  if [[ -f "/var/spool/cron/$(whoami)" ]];

  then

   CRON\_DIR\_PATH="/var/spool/cron/${USERNAME}"

  else

   error "No available cron found. "

  fi

 fi

}

function send\_alarm\_to\_tty()

{

 for tty in `find /dev/pts -user ${USERNAME} -group tty -type c`

 do

  echo -e "$1" >> $tty

 done

}

function add\_alarm\_command\_to\_cron()

{

 local date=$1

 local month=$2

 local day=$3

 local hour=$4

 local min=$5

 command=$(cat <<-EOF

*## CONTACT:${USERNAME}:${date} START##*

${min} ${hour} ${day} ${month} \* ${SCRIPTPATH} do\_alarm ${date} 2>&1 1>/dev/null

*## CONTACT:${USERNAME}:${date} END##*

EOF

)

 if ! [[ -d "${CRON\_DIR\_PATH}" ]];

 then

  error "Cron not available on this system"

  return 0

 fi

 if ! [[ -e "${CRON\_DIR\_PATH}/${USERNAME}" ]];

 then

  touch "${CRON\_DIR\_PATH}/${USERNAME}"

*# make cron happy*

  chmod 600 "${CRON\_DIR\_PATH}/${USERNAME}"

 fi

 echo "$command" >> "${CRON\_DIR\_PATH}/${USERNAME}"

 return 1

}

*# Setup database*

function db\_setup()

{

*# Check database directory*

 DB\_DIR\_PATH=$(realpath "${DB\_DIR\_PATH}")

 info "Database directory is ${DB\_DIR\_PATH}"

 if ! [[ -d ${DB\_DIR\_PATH} ]];

 then

  [[ -e ${DB\_DIR\_PATH} ]] && error "${DB\_DIR\_PATH} is not a directory" && exit $EXIT\_FAILURE

*# If not exists, create one*

  mkdir -p ${DB\_DIR\_PATH}

  chmod 777 ${DB\_DIR\_PATH}

 fi

*# CHeck database*

 DB\_FILE\_PATH="${DB\_DIR\_PATH}/${USERNAME}.db"

 info "Database path is ${DB\_FILE\_PATH}"

 if ! [[ -f ${DB\_FILE\_PATH} ]];

 then

  [[ -e ${DB\_FILE\_PATH} ]] && error "${DB\_FILE\_PATH} is not a file" && exit $EXIT\_FAILURE

*# If not exists, create one*

  touch ${DB\_FILE\_PATH}

  chmod 600 ${DB\_FILE\_PATH}

 fi

}

*# Load database*

function db\_load() {

*# Database file format*

*## 1. Each line represents a contact entry.*

*## 2. Each contact entry contains eleven fields: date, alarm, name, group, priority, phone, wechat, qq, email, company, note.*

*## 3. date is the timestamp when the contact has created.*

*## 4. alarm is the timestamp when the alarm for this contact will be sent,*

*##   if the value is 0, it means the alarm is disabled.*

*## 5. title and context are base64 encoded.*

 CONTACT\_DATE=($(cut -f1 -d":" $DB\_FILE\_PATH))

 CONTACT\_ALARM=($(cut -f2 -d":" $DB\_FILE\_PATH))

 CONTACT\_NAME=($(cut -f3 -d":" $DB\_FILE\_PATH))

 CONTACT\_GROUP=($(cut -f4 -d":" $DB\_FILE\_PATH))

 CONTACT\_PRI=($(cut -f5 -d":" $DB\_FILE\_PATH))

 CONTACT\_PHONE=($(cut -f6 -d":" $DB\_FILE\_PATH))

 CONTACT\_WECHAT=($(cut -f7 -d":" $DB\_FILE\_PATH))

 CONTACT\_QQ=($(cut -f8 -d":" $DB\_FILE\_PATH))

 CONTACT\_EMAIL=($(cut -f9 -d":" $DB\_FILE\_PATH))

 CONTACT\_COMPANY=($(cut -f10 -d":" $DB\_FILE\_PATH))

 CONTACT\_NOTE=($(cut -f11 -d":" $DB\_FILE\_PATH))

 CONTACT\_LENGTH=${#CONTACT\_DATE[@]}

}

*# Add an entry to database*

function db\_add() {

 local date=$(date +%s)

 local alarm=0

 local name=$1

 local group=$2

 local priority=$3

 local phone=$4

 local wechat=$5

 local qq=$6

 local email=$7

 local company=$8

 local note=$9

 CONTACT\_DATE[$CONTACT\_LENGTH]=$date

 CONTACT\_ALARM[$CONTACT\_LENGTH]=$alarm

 CONTACT\_NAME[$CONTACT\_LENGTH]=$name

 CONTACT\_GROUP[$CONTACT\_LENGTH]=$group

 CONTACT\_PRI[$CONTACT\_LENGTH]=$priority

 CONTACT\_PHONE[$CONTACT\_LENGTH]=$phone

 CONTACT\_WECHAT[$CONTACT\_LENGTH]=$wechat

 CONTACT\_QQ[$CONTACT\_LENGTH]=$qq

 CONTACT\_EMAIL[$CONTACT\_LENGTH]=$email

 CONTACT\_COMPANY[$CONTACT\_LENGTH]=$company

 CONTACT\_NOTE[$CONTACT\_LENGTH]=$note

 ((CONTACT\_LENGTH++))

}

*# Remove an entry to database*

function db\_del() {

 local i=$1

*# Check if index is OK*

 [[ $i -lt 0 || $i -ge ${CONTACT\_LENGTH} ]] && return

*# Are we removing an entry located in the end of array?*

 if [[ $i == $((${CONTACT\_LENGTH}-1)) ]]

 then

  unset CONTACT\_DATE[$i]

  unset CONTACT\_ALARM[$i]

  unset CONTACT\_NAME[$i]

  unset CONTACT\_GROUP[$i]

  unset CONTACT\_PRI[$i]

  unset CONTACT\_PHONE[$i]

  unset CONTACT\_WECHAT[$i]

  unset CONTACT\_QQ[$i]

  unset CONTACT\_EMAIL[$i]

  unset CONTACT\_COMPANY[$i]

  unset CONTACT\_NOTE[$i]

*# Are we removing an entry located in the head of array?*

 elif [[ $i == 0 ]]

 then

*# Copy the entire array except the first element*

  CONTACT\_DATE=(${CONTACT\_DATE[@]:1})

  CONTACT\_ALARM=(${CONTACT\_ALARM[@]:1})

  CONTACT\_NAME=(${CONTACT\_NAME[@]:1})

  CONTACT\_GROUP=(${CONTACT\_GROUP[@]:1})

  CONTACT\_PRI=(${CONTACT\_PRI[@]:1})

  CONTACT\_PHONE=(${CONTACT\_PHONE[@]:1})

  CONTACT\_WECHAT=(${CONTACT\_WECHAT[@]:1})

  CONTACT\_QQ=(${CONTACT\_QQ[@]:1})

  CONTACT\_EMAIL=(${CONTACT\_EMAIL[@]:1})

  CONTACT\_COMPANY=(${CONTACT\_COMPANY[@]:1})

  CONTACT\_NOTE=(${CONTACT\_NOTE[@]:1})

*# Are we removing an entry located in the middle of array?*

 else

*# S1: Copy all the elements ahead of the entry to new array*

  local date=${CONTACT\_DATE[@]:0:$i}

*# S2: Append all the elements behind of the entry to new array*

  date+=(${CONTACT\_DATE[@]:(($i+1))})

*# S3: Replace the old array with the new one*

  CONTACT\_DATE=(${date[@]})

  local alarm=${CONTACT\_ALARM[@]:0:$i}

  alarm+=(${CONTACT\_ALARM[@]:(($i+1))})

  CONTACT\_ALARM=(${alarm[@]})

  local name=${CONTACT\_NAME[@]:0:$i}

  name+=(${CONTACT\_NAME[@]:(($i+1))})

  CONTACT\_NAME=(${name[@]})

  local group=${CONTACT\_GROUP[@]:0:$i}

  group+=(${CONTACT\_GROUP[@]:(($i+1))})

  CONTACT\_GROUP=(${group[@]})

  local priority=${CONTACT\_PRI[@]:0:$i}

  priority+=(${CONTACT\_PRI[@]:(($i+1))})

  CONTACT\_PRI=(${priority[@]})

  local phone=${CONTACT\_PHONE[@]:0:$i}

  phone+=(${CONTACT\_PHOEN[@]:(($i+1))})

  CONTACT\_PHONE=(${phone[@]})

  local wechat=${CONTACT\_WECHAT[@]:0:$i}

  wechat+=(${CONTACT\_WECHAT[@]:(($i+1))})

  CONTACT\_WECHAT=(${wechat[@]})

  local qq=${CONTACT\_QQ[@]:0:$i}

  qq+=(${CONTACT\_QQ[@]:(($i+1))})

  CONTACT\_QQ=(${qq[@]})

  local email=${CONTACT\_EMAIL[@]:0:$i}

  email+=(${CONTACT\_EMAIL[@]:(($i+1))})

  CONTACT\_EMAIL=(${email[@]})

  local company=${CONTACT\_COMPANY[@]:0:$i}

  company+=(${CONTACT\_COMPANY[@]:(($i+1))})

  CONTACT\_COMPANY=(${company[@]})

  local note=${CONTACT\_NOTE[@]:0:$i}

  note+=(${CONTACT\_NOTE[@]:(($i+1))})

  CONTACT\_NOTE=(${note[@]})

 fi

 ((CONTACT\_LENGTH--))

}

function db\_save() {

*# Delete all the data in the database file*

 > ${DB\_FILE\_PATH}

*# Write entries in the array line by line to the database*

 for (( i=0; i < ${CONTACT\_LENGTH}; i++ )); do

  local e="${CONTACT\_DATE[$i]}:${CONTACT\_ALARM[$i]}:${CONTACT\_NAME[$i]}:${CONTACT\_GROUP[$i]}:${CONTACT\_PRI[$i]}:${CONTACT\_PHONE[$i]}:${CONTACT\_WECHAT[$i]}:${CONTACT\_QQ[$i]}:${CONTACT\_EMAIL[$i]}:${CONTACT\_COMPANY[$i]}:${CONTACT\_NOTE[$i]}"

  echo $e >> ${DB\_FILE\_PATH}

 done

}

function contact\_add() {

 local name\_en=$(echo -n "$1" | base64 -w0)

 local group\_en=$(echo -n "$2" | base64 -w0)

 local priority\_en=$(echo -n "$3" | base64 -w0 )

 local phone\_en=$(echo -n "$4" | base64 -w0)

 local wechat\_en=$(echo -n "$5" | base64 -w0)

 local qq\_en=$(echo -n "$6" | base64 -w0)

 local email\_en=$(echo -n "$7" | base64 -w0 )

 local company\_en=$(echo -n "$8" | base64 -w0)

 local note\_en=$(echo -n "$9" | base64 -w0)

 db\_add $name\_en $group\_en $priority\_en $phone\_en $wechat\_en $qq\_en $email\_en $company\_en $note\_en

}

function contact\_del()

{

 local i=$1

 [[ $i -lt 1 || $i -gt ${CONTACT\_LENGTH} ]] && return

 db\_del $(($i-1))

}

function contact\_list()

{

 for (( i=0; i < ${CONTACT\_LENGTH}; i++ ));

 do

  local date=$(date -d@"${CONTACT\_DATE[$i]}" '+%Y/%m/%d %H:%M:%S')

  local name=$(echo -n "${CONTACT\_NAME[$i]}" | base64 -d -w0)

  if [[ ${CONTACT\_ALARM[$i]} == "0" ]];

  then

   local ala="NO "

  else

   local ala="YES"

  fi

  echo -e "$((i+1)). \033[32;1mDate\033[0m: $date \033[32;1mAlarm\033[0m: $ala \033[32;1mTitle\033[0m: $name"

 done

}

function init()

{

 print\_title

 get\_username

 get\_cron

 db\_setup

 info "Loading database..."

 db\_load

 info "Loaded ${CONTACT\_LENGTH} entries from database."

 info "Initialization done!"

 waituser

}

function add()

{

 print\_title

 echo -n "Please input the name: "

 espeak\_w "Please input the name."

 read name

 echo -n "Please input the group: "

 espeak\_w "Please input the group."

 read group

 echo -n "Please input the priority: "

 espeak\_w "Please input the priority."

 read priority

 echo -n "Please input the phone number: "

 espeak\_w "Please input the phone number."

 read phone

 echo -n "Please input the wechat: "

 espeak\_w "Please input the wechat."

 read wechat

 echo -n "Please input the qq number: "

 espeak\_w "Please input the qq number."

 read qq

 echo -n "Please input the email: "

 espeak\_w "Please input the email."

 read email

 echo -n "Please input the company: "

 espeak\_w "Please input the company."

 read company

 echo "Please input the notes (press [Ctrl+D] to finish):"

 espeak\_w "Please input the notes .Attention press [Ctrl+D] to finish."

 local note=$(< /dev/stdin)

 contact\_add "$name" "$group" "$priority" "$phone" "$wechat" "$qq" "$email" "$company" "$note"

 echo

 info "New Contact added!"

 espeak\_w "New Contact added."

}

function edit()

{

 print\_title

 if [[ ${CONTACT\_LENGTH} == 0 ]];

 then

  info "There is no contacts stored in your database!"

  espeak\_w "There is no contacts stored in your database."

  return

 fi

 echo "Please select the contact you want to edit from the following list:"

 espeak\_w "Please select the contact you want to edit from the following list."

 contact\_list

 if [[ ${CONTACT\_LENGTH} == 1 ]];

 then

  option="1"

 else

  echo "You can use 0 to go back to the main menu. "

  echo -e -n "\nPlease input the index [1-${CONTACT\_LENGTH}]: "

  espeak\_w "Please input the index. Use 0 to go back. "

  read option

 fi

 if [[ $option == "Q" || $option == "q" || $option == "0" ]];

 then

  echo "Going back to the main menu. "

  espeak\_w "Going back. "

  return

 fi

 if ! [[ $option =~ $NUMREGEX ]];

 then

  error "Please input a vaill number!"

  espeak\_w "Please input a vaill number."

  return

 fi

 if ! [[ $option > 0 && $option < $((CONTACT\_LENGTH+1)) ]];

 then

  error "Index Out of Range [1-${CONTACT\_LENGTH}]!"

  espeak\_w "Index Out of Range."

  return

 fi

 echo -e "\nSelected Contact: $option"

 espeak\_w "Please select a Contact."

 ((opt--))

 echo -n "Do you want to edit the name? [Y/N]: "

 espeak\_w "Do you want to edit the name?Y for Yes while N for No."

 read confirm

 if [[ "$confirm" == "Y" || "$confirm" == "y" ]];

 then

  local oldname=$(echo -n "${CONTACT\_NAME[$option]}" | base64 -d -w0)

  echo -e "\033[32;1mOld Name:\033[0m $oldname"

  espeak\_w "The old name is $oldname."

  echo -ne "\033[32;1mNew Name:\033[0m "

  espeak\_w "Please input the new name."

  read newname

  CONTACT\_NAME[$option]=$(echo -n "$newname" | base64 -w0)

 fi

 echo -n "Do you want to edit the group? [Y/N]: "

 espeak\_w "Do you want to edit the group?Y for Yes while N for No."

 read confirm

 if [[ "$confirm" == "Y" || "$confirm" == "y" ]];

 then

  local oldgroup=$(echo -n "${CONTACT\_GROUP[$option]}" | base64 -d -w0)

  echo -e "\033[32;1mOld Group:\033[0m $oldgroup"

  espeak\_w "The old group is $oldgroup."

  echo -ne "\033[32;1mNew Group:\033[0m "

  espeak\_w "Please input the new group."

  read newgroup

  CONTACT\_GROUP[$option]=$(echo -n "$newgroup" | base64 -w0)

 fi

 echo -n "Do you want to edit the priority? [Y/N]: "

 espeak\_w "Do you want to edit the priority?Y for Yes while N for No."

 read confirm

 if [[ "$confirm" == "Y" || "$confirm" == "y" ]];

 then

  local oldpriority=$(echo -n "${CONTACT\_PRI[$option]}" | base64 -d -w0)

  echo -e "\033[32;1mOld Priority:\033[0m $oldpriority"

  espeak\_w "The old priority is $oldpriority."

  echo -ne "\033[32;1mNew Priority:\033[0m "

  espeak\_w "Please input the new priority."

  read newpriority

  CONTACT\_PRI[$option]=$(echo -n "$newpriority" | base64 -w0)

 fi

 echo -n "Do you want to edit the phone number? [Y/N]: "

 espeak\_w "Do you want to edit the phone number?Y for Yes while N for No."

 read confirm

 if [[ "$confirm" == "Y" || "$confirm" == "y" ]];

 then

  local oldphone=$(echo -n "${CONTACT\_PHONE[$option]}" | base64 -d -w0)

  echo -e "\033[32;1mOld Phone:\033[0m $oldphone"

  espeak\_w "The old phone is $oldphone."

  echo -ne "\033[32;1mNew Phone:\033[0m "

  espeak\_w "Please input the new phone."

  read newphone

  CONTACT\_PHONE[$option]=$(echo -n "$newphone" | base64 -w0)

 fi

 echo -n "Do you want to edit the wechat? [Y/N]: "

 espeak\_w "Do you want to edit the wehcat?Y for Yes while N for No."

 read confirm

 if [[ "$confirm" == "Y" || "$confirm" == "y" ]];

 then

  local oldwechat=$(echo -n "${CONTACT\_WECHAT[$option]}" | base64 -d -w0)

  echo -e "\033[32;1mOld Wechat:\033[0m $oldwechat"

  espeak\_w "The old name is $oldwechat."

  echo -ne "\033[32;1mNew Wechat:\033[0m "

  espeak\_w "Please input the new wechat."

  read newwechat

  CONTACT\_WECHAT[$option]=$(echo -n "$newwechat" | base64 -w0)

 fi

 echo -n "Do you want to edit the qq? [Y/N]: "

 espeak\_w "Do you want to edit the qq?Y for Yes while N for No."

 read confirm

 if [[ "$confirm" == "Y" || "$confirm" == "y" ]];

 then

  local oldqq=$(echo -n "${CONTACT\_QQ[$option]}" | base64 -d -w0)

  echo -e "\033[32;1mOld QQ:\033[0m $oldqq"

  espeak\_w "The old qq is $oldqq."

  echo -ne "\033[32;1mNew QQ:\033[0m "

  espeak\_w "Please input the new qq."

  read newqq

  CONTACT\_QQ[$option]=$(echo -n "$newqq" | base64 -w0)

 fi

 echo -n "Do you want to edit the email? [Y/N]: "

 espeak\_w "Do you want to edit the email?Y for Yes while N for No."

 read confirm

 if [[ "$confirm" == "Y" || "$confirm" == "y" ]];

 then

  local oldmail=$(echo -n "${CONTACT\_EMAIL[$option]}" | base64 -d -w0)

  echo -e "\033[32;1mOld Email:\033[0m $oldmail"

  espeak\_w "The old Email is $oldmail."

  echo -ne "\033[32;1mNew Email:\033[0m "

  espeak\_w "Please input the new Email."

  read newemail

  CONTACT\_EMAIL[$option]=$(echo -n "$newemail" | base64 -w0)

 fi

 echo -n "Do you want to edit the company? [Y/N]: "

 espeak\_w "Do you want to edit the company?Y for Yes while N for No."

 read confirm

 if [[ "$confirm" == "Y" || "$confirm" == "y" ]];

 then

  local oldcompany=$(echo -n "${CONTACT\_COMPANY[$option]}" | base64 -d -w0)

  echo -e "\033[32;1mOld Company:\033[0m $oldcompany"

  espeak\_w "The old company is $oldcompany."

  echo -ne "\033[32;1mNew Company:\033[0m "

  espeak\_w "Please input the new company."

  read newcompany

  CONTACT\_COMPANY[$option]=$(echo -n "$newcompany" | base64 -w0)

 fi

 echo -n "Do you want to edit the note? [Y/N]: "

 espeak\_w "Do you want to edit the note?Y for Yes while N for No."

 read confirm

 if [[ "$confirm" == "Y" || "$confirm" == "y" ]];

 then

  local oldnote=$(echo -n "${CONTACT\_NOTE[$option]}" | base64 -d -w0)

  echo -e "\033[32;1mOld Note:\033[0m \n$oldnote"

  espeak\_w "The ole note is $oldnote."

  echo -e "\033[32;1mNew Note:\033[0m (press [Ctrl+D to finish]"

  espeak\_w "Please input a new note.Attention (press [Ctrl+D to finish]."

  local newnote=$(< /dev/stdin)

  CONTACT\_NOTE[$option]=$(echo -n "$newnote" | base64 -w0)

 fi

 echo

 info "Done!"

}

function del()

{

 print\_title

 if [[ ${CONTACT\_LENGTH} == 0 ]];

 then

  info "There is no contacts stored in your database!"

  espeak\_w "There is no contacts stored in your database!"

  return

 fi

 echo "Please select the contact you want to delete from the following list: "

 espeak\_w "Please select the contact you want to delete from the following list. "

 contact\_list

 if [[ ${CONTACT\_LENGTH} == 1 ]];

 then

  option="1"

 else

  echo "You can use 0 to go back to the main menu. "

  echo -e -n "\nPlease input the index [1-${CONTACT\_LENGTH}]: "

  espeak\_w "Please input the index. Use 0 to go back. "

  read option

 fi

 if [[ $option == "Q" || $option == "q" || $option == "0" ]];

 then

  echo "Going back to the main menu. "

  espeak\_w "Going back. "

  return

 fi

 if ! [[ $option =~ $NUMREGEX ]];

 then

  error "Please input a vaill number!"

  espeak\_w "Please input a vaill number."

  return

 fi

 if ! [[ $option > 0 && $option < $((CONTACT\_LENGTH+1)) ]];

 then

  error "Index Out of Range [1-${CONTACT\_LENGTH}]!"

  espeak\_w "Index Out of Range ."

  return

 fi

 echo -e "\nSelected Contact: $option"

 espeak\_w "Please select a Contact."

 echo -e "\033[31;1mDO YOU REALLY WANT TO DELETE THIS CONTACT?\033[0m"

 espeak\_w "DO YOU REALLY WANT TO DELETE THIS CONTACT?"

 echo -n "[Input YES to continue, else to abort]: "

 espeak\_w "Input YES to continue, else to abort. "

 read confirm

 if [[ $confirm == "YES" ]];

 then

  contact\_del $option

  echo "Done!"

  espeak\_w "Done."

 else

  echo "Aborted."

  espeak\_w "Aborted."

 fi

}

function alarm()

{

 print\_title

 if [[ ${CONTACT\_LENGTH} == 0 ]];

 then

  info "Sorry, there is no contacts stored in your database! "

  espeak\_w "Sorry, there is no contacts stored in your database! "

  return

 fi

 echo "Please select the contact you want to set/cancel alarm from the following list: "

 espeak\_w "Please select the contact you want to set or cancel alarm from the following list. "

 contact\_list

 if [[ ${CONTACT\_LENGTH} == 1 ]];

 then

  option="1"

 else

  echo "You can use 0 to go back to the main menu. "

  echo -en "\nPlease input the index [1-${CONTACT\_LENGTH}]: "

  espeak\_w "Please input the index. Use 0 to go back. "

  read option

 fi

 if [[ $option == "Q" || $option == "q" || $option == "0" ]];

 then

  echo "Going back to the main menu. "

  espeak\_w "Going back. "

  return

 fi

 if ! [[ $option =~ $NUMREGEX ]];

 then

  error "Please input a vaill number!"

  espeak\_w "Please input a vaill number."

  return

 fi

 if ! [[ $option > 0 && $option < $((CONTACT\_LENGTH+1)) ]];

 then

  error "Index Out of Range [1-${CONTACT\_LENGTH}]!"

  espeak\_w "Index Out of Range."

  return

 fi

 echo -e "\nSelected Contact: $option"

 ((opt--))

 if [[ ! ${CONTACT\_ALARM[$option]} == '0' ]];

 then

  CONTACT\_ALARM[$option]='0'

  ((opt++)) *# Related to the index shown*

  info "Selected Contact $option alarm is now cancelled. "

  espeak\_w "Selected Contact $option alarm is now cancelled. "

  return

 fi

 espeak\_w "Please select a contact. "

 echo -e "Set an alarm to remind yourself to call back! "

 espeak\_w "Set an alarm to remind yourself to call back! "

 echo -e "The format is as flollows: "

 espeak\_w "The format is as flollows. "

 echo -e "Month/Day Hour:Minute"

 espeak\_w "Month/Day Hour:Minute"

 echo -e "Note that this is a 24-hour schedule"

 espeak\_w "Note that this is a 24-hour schedule"

 echo -ne "Input: "

 read alarm\_date

 local alarm\_ts=$(date -d"$(date +%Y)/$alarm\_date:00" +%s)

 if [[ -z "$alarm\_ts" ]];

 then

  error "$alarm\_date is not a vaild date! "

  espeak\_w "$alarm\_date is not a vaild date! "

  return

 fi

 local now=$(date +%s)

 if [[ (( alarm\_ts -le now)) ]];

 then

  error "Alarm date should not be earlier than present time. "

  espeak\_w "Alarm date should not be earlier than present time. "

  return

 fi

 local month=$(date -d@"$alarm\_ts" +%m)

 local day=$(date -d@"$alarm\_ts" +%d)

 local hour=$(date -d@"$alarm\_ts" +%H)

 local min=$(date -d@"$alarm\_ts" +%M)

 local date=${CONTACT\_DATE[$option]}

 add\_alarm\_command\_to\_cron $date $month $day $hour $min

 if [[ $? -eq 1 ]];

 then

  CONTACT\_ALARM[$option]=$alarm\_ts

  echo "Done! Your contact will be alarmed at $(date -d@"$alarm\_ts" '+%Y/%m/%d %H:%M:%S')"

  espeak\_w "Done! Your contact will be alarmed at $(date -d@"$alarm\_ts" '+%Y/%m/%d %H:%M:%S')"

 fi

}

function list()

{

 print\_title

 if [[ ${CONTACT\_LENGTH} == 0 ]];

 then

  info "There is no contacts stored in your database!"

  espeak\_w "There is no contacts stored in your database!"

  return

 fi

 echo "Available Contact:"

 espeak\_w "Available Contact is as followed."

 contact\_list

 if [[ ${CONTACT\_LENGTH} == 1 ]];

 then

  echo -en "\nPlease input 1 to view contact context, or Q to return to menu: "

  espeak\_w "Please input 1 to view contact context, or Q to return to menu."

 else

  echo -en "\nPlease input [1-${CONTACT\_LENGTH}] to view contact context, or Q to return to menu: "

  espeak\_w "Please input [1-${CONTACT\_LENGTH}] to view contact context, or Q to return to menu."

 fi

 read option

 while :

 do

  if [[ $option == "Q" || $option == "q" || $option == "0" ]];

  then

   return

  elif ! [[ $option =~ $NUMREGEX ]];

  then

   error "Please input a vaill number!"

   espeak\_w "Please input a vaill number."

  elif ! [[ $option > 0 && $option < $((CONTACT\_LENGTH+1)) ]];

  then

   error "Index Out of Range [1-${CONTACT\_LENGTH}]!"

   espeak\_w "Index out of range."

  else

   print\_title

   ((opt--))

   local date=$(date -d@"${CONTACT\_DATE[$option]}" '+%Y/%m/%d %H:%M:%S')

   local name=$(echo -n "${CONTACT\_NAME[$option]}" | base64 -d -w0)

   local group=$(echo -n "${CONTACT\_GROUP[$option]}" | base64 -d -w0)

   local priority=$(echo -n "${CONTACT\_PRI[$option]}" | base64 -d -w0)

   local phone=$(echo -n "${CONTACT\_PHONE[$option]}" | base64 -d -w0)

   local wechat=$(echo -n "${CONTACT\_WECHAT[$option]}" | base64 -d -w0)

   local qq=$(echo -n "${CONTACT\_QQ[$option]}" | base64 -d -w0)

   local email=$(echo -n "${CONTACT\_EMAIL[$option]}" | base64 -d -w0)

   local company=$(echo -n "${CONTACT\_COMPANY[$option]}" | base64 -d -w0)

   local note=$(echo -n "${CONTACT\_NOTE[$option]}" | base64 -d -w0)

   echo -e "\033[32;1mDate\033[0m: $date"

   espeak\_w "Date: $date."

   echo -e "\033[32;1mName\033[0m: $name"

   espeak\_w "Name:$name."

   echo -e "\033[32;1mGroup\033[0m: $group"

   espeak\_w "Group:$group."

   echo -e "\033[32;1mPriority\033[0m: $priority"

   espeak\_w "Priority:$priority"

   echo -e "\033[32;1mPhone\033[0m: $phone"

   espeak\_w "Phone:$phone."

   echo -e "\033[32;1mWechat\033[0m: $wechat"

   espeak\_w "Wechat:$wechat."

   echo -e "\033[32;1mQQ\033[0m: $qq"

   espeak\_w "QQ:$qq"

   echo -e "\033[32;1mEmail\033[0m: $email"

   espeak\_w "Email:$email."

   echo -e "\033[32;1mCompany\033[0m: $company"

   espeak\_w "Company:$company."

   echo -e "\033[32;1mNote\033[0m: \n$note\n"

   espeak\_w "Note:$note"

   waituser

   list

   return

  fi

  echo -n "Input:"

  read option

 done

}

function exit\_program()

{

 clear

 info "Saving database..."

 db\_save

 info "Saved ${CONTACT\_LENGTH} entries to database."

 echo -e "\n\033[32;1mThank you for using Universal Contact. \nWish you a better tomorrow! \033[0m"

 espeak\_w "Thanks for using our program. Have a nice day!"

 exit $EXIT\_SUCCESS

}

function menu()

{

 print\_title

 echo -e "\033[32;1mWelcome, ${USERNAME}! \033[0m\n"

 espeak\_w "Welcome, ${USERNAME}! "

 echo "Available Options:  "

 echo -e "\t1 = \033[32m[A]\033[0mdd Contact"

 echo -e "\t2 = \033[32m[D]\033[0melete Contact"

 echo -e "\t3 = \033[32m[E]\033[0mdit Contact"

 echo -e "\t4 = \033[32m[L]\033[0mist Contact"

 echo -e "\t5 = \033[32m[S]\033[0met Alarm for a Contact"

 echo -e "\t0 = \033[32m[Q]\033[0muit this program"

 echo -ne "\nPlease select an option to continue [A/D/E/L/S/Q]: "

 espeak\_w "Please select an option."

 read option

 case $option in

  "1" | "A" | "a")

   add

  ;;

  "2" | "D" | "d")

   del

  ;;

  "3" | "E" | "e")

   edit

  ;;

  "4" | "L" | "l")

   list

  ;;

  "5" | "S" | "s")

   alarm

  ;;

  "0" | "Q" | "q")

   exit\_program

  ;;

  \*)

   error "Unknow option!"

   espeak\_w "Unkow option."

  ;;

 esac

 waituser

}

function add\_command()

{

 if [[ $ARGC < 3 ]];

 then

  echo "USAGE ${ADD\_HELP}"

  return

 fi

 local name=${ARGV[1]}

 local group=${ARGV[2]}

 local priority=${ARGV[3]}

 local phone=${ARGV[4]}

 local wechat=${ARGV[5]}

 local qq=${ARGV[6]}

 local email=${ARGV[7]}

 local company=${ARGV[8]}

 local note=${ARGV[9]}

 db\_load

 contact\_add "$name" "$group" "$priority" "$phone" "$wechat" "$qq" "$email" "$company" "$note"

 db\_save

 espeak\_w "Please input the name: "

 echo -e "\033[32;1mName\033[0m: $name"

 espeak\_w "Please input the group: "

 echo -e "\033[32;1mGroup\033[0m: $group"

 espeak\_w "Please input the priority: "

 echo -e "\033[32;1mPriority\033[0m: $priority"

 espeak\_w "Please input the phone: "

 echo -e "\033[32;1mPhone\033[0m: $phone"

 espeak\_w "Please input the wechat: "

 echo -e "\033[32;1mWechat\033[0m: $wechat"

 espeak\_w "Please input the QQ: "

 echo -e "\033[32;1mQQ\033[0m: $qq"

 espeak\_w "Please input the Email: "

 echo -e "\033[32;1mEmail\033[0m: $email"

 espeak\_w "Please input the company: "

 echo -e "\033[32;1mCompany\033[0m: $company"

 espeak\_w "Please input the note: "

 echo -e "\033[32;1mNote\033[0m: \n$note\n"

 info "New Contact added!"

 espeak\_w "New Contact added! "

}

function edit\_command()

{

 if [[ $ARGC < 4 ]];

 then

  echo "USAGE ${EDIT\_HELP}"

  return

 fi

 local idx=${ARGV[1]}

 local type=${ARGV[2]}

 local new=${ARGV[3]}

 db\_load

 if [[ ${CONTACT\_LENGTH} == 0 ]];

 then

  info "There is no contacts stored in your database!"

  espeak\_w "There is no contacts stored in your database."

  return

 fi

 if ! [[ $idx =~ $NUMREGEX ]];

 then

  error "$idx is not a vaill number!"

  espeak\_w "$idx is not a vaill number."

  return

 fi

 if ! [[ $idx > 0 && $idx < $((CONTACT\_LENGTH+1)) ]];

 then

  error "Index Out of Range [1-${CONTACT\_LENGTH}]!"

  espeak\_w "Index Out of Range."

  return

 fi

 ((idx--))

 case $type in

  "name")

   CONTACT\_NAME[$idx]=$(echo -n "$new" | base64 -w0)

  ;;

  "group")

   CONTACT\_CONTEXT[$idx]=$(echo -n "$new" | base64 -w0)

  ;;

  "priority")

   CONTACT\_PRI[$idx]=$(echo -n "$new" | base64 -w0)

  ;;

  "phone")

   CONTACT\_PHONE[$idx]=$(echo -n "$new" | base64 -w0)

  ;;

  "wechat")

   CONTACT\_WECHAT[$idx]=$(echo -n "$new" | base64 -w0)

  ;;

  "qq")

   CONTACT\_QQ[$idx]=$(echo -n "$new" | base64 -w0)

  ;;

  "email")

   CONTACT\_EMAIL[$idx]=$(echo -n "$new" | base64 -w0)

  ;;

  "company")

   CONTACT\_COMPANY[$idx]=$(echo -n "$new" | base64 -w0)

  ;;

  "note")

   CONTACT\_NOTE[$idx]=$(echo -n "$new" | base64 -w0)

  ;;

  \*)

   error "Unknow type!"

  ;;

 esac

 db\_save

 echo "Done!"

}

function del\_command()

{

 if [[ $ARGC < 2 ]];

 then

  echo "USAGE ${DEL\_HELP}"

  return

 fi

 local idx=${ARGV[1]}

 db\_load

 if [[ ${CONTACT\_LENGTH} == 0 ]];

 then

  info "There is no contacts stored in your database!"

  espeak\_w "There is no contacts stored in your database."

  return

 fi

 if ! [[ $idx =~ $NUMREGEX ]];

 then

  error "$idx is not a vaill number!"

  espeak\_w "$idx is not a vaill number!"

  return

 fi

 if ! [[ $idx > 0 && $idx < $((CONTACT\_LENGTH+1)) ]];

 then

  error "Index Out of Range [1-${CONTACT\_LENGTH}]!"

  espeak\_w "Index Out of Range."

  return

 fi

 contact\_del $idx

 db\_save

 echo "Done!"

}

function list\_command()

{

 if [[ $ARGC < 2 ]];

 then

  db\_load

  if [[ ${CONTACT\_LENGTH} == 0 ]];

  then

   info "There is no contacts stored in your database!"

   espeak\_w "There is no contacts stored in your database."

   return

  fi

  echo "Available Contact:"

  espeak\_w "Available Contact."

  contact\_list

 else

  local idx=${ARGV[1]}

  db\_load

  if [[ ${CONTACT\_LENGTH} == 0 ]];

  then

   info "There is no contacts stored in your database!"

   espeak\_w "There is no contacts stored in your database!"

   return

  fi

  if ! [[ $idx =~ $NUMREGEX ]];

  then

   error "$idx is not a vaill number!"

   espeak\_w "$idx is not a vaill number!"

   return

  fi

  if ! [[ $idx > 0 && $idx < $((CONTACT\_LENGTH+1)) ]]; then

   error "Index Out of Range [1-${CONTACT\_LENGTH}]!"

   espeak\_w "Index Out of Range [1-${CONTACT\_LENGTH}]!"

   return

  fi

  ((idx--))

  local date=$(date -d@"${CONTACT\_DATE[$idx]}" '+%Y/%m/%d %H:%M:%S')

  local name=$(echo -n "${CONTACT\_NAME[$idx]}" | base64 -d -w0)

  local group=$(echo -n "${CONTACT\_GROUP[$idx]}" | base64 -d -w0)

  local priority=$(echo -n "${CONTACT\_PRI[$idx]}" | base64 -d -w0)

  local phone=$(echo -n "${CONTACT\_PHONE[$idx]}" | base64 -d -w0)

  local wechat=$(echo -n "${CONTACT\_WECHAT[$idx]}" | base64 -d -w0)

  local qq=$(echo -n "${CONTACT\_QQ[$idx]}" | base64 -d -w0)

  local email=$(echo -n "${CONTACT\_EMAIL[$idx]}" | base64 -d -w0)

  local company=$(echo -n "${CONTACT\_COMPANY[$idx]}" | base64 -d -w0)

  local note=$(echo -n "${CONTACT\_NOTE[$idx]}" | base64 -d -w0)

  echo -e "\033[32;1mDate\033[0m: $date"

  espeak\_w "Date:$date"

  echo -e "\033[32;1mName\033[0m: $name"

  espeak\_w "Name:$name."

  echo -e "\033[32;1mGroup\033[0m: $group"

  espeak\_w "Group:$group."

  echo -e "\033[32;1mPriority\033[0m: $priority"

  espeak\_w "Priority:$priority"

  echo -e "\033[32;1mPhone\033[0m: $phone"

  espeak\_w "Phone:$phone."

  echo -e "\033[32;1mWechat\033[0m: $wechat"

  espeak\_w "Wechat:$wechat."

  echo -e "\033[32;1mQQ\033[0m: $qq"

  espeak\_w "QQ:$qq"

  echo -e "\033[32;1mEmail\033[0m: $email"

  espeak\_w "Email:$email."

  echo -e "\033[32;1mCompany\033[0m: $company"

  espeak\_w "Company:$company."

  echo -e "\033[32;1mNote\033[0m: \n$note\n"

  espeak\_w "Note:$note"

 fi

}

function alarm\_command()

{

 if [[ $ARGC < 2 ]];

 then

  echo "USAGE ${ALARM\_HELP}"

  return

 fi

 local idx=${ARGV[1]}

 local alarm\_date=${ARGV[2]}

 db\_load

 if [[ ${CONTACT\_LENGTH} == 0 ]];

 then

  info "There is no contacts stored in your database!"

  espeak\_w "There is no contacts stored in your database!"

  return

 fi

 if ! [[ $idx =~ $NUMREGEX ]];

 then

  error "$idx is not a vaill number!"

  espeak\_w "$idx is not a vaill number!"

  return

 fi

 if ! [[ $idx > 0 && $idx < $((CONTACT\_LENGTH+1)) ]];

 then

  error "Index Out of Range [1-${CONTACT\_LENGTH}]!"

  espeak\_w "Index Out of Range."

  return

 fi

 ((idx--))

 local date=${CONTACT\_DATE[$idx]}

 IFS='-' read -ra alarm\_date <<< "$alarm\_date"

 if [[ ${#alarm\_date[@]} == 2 ]];

 then

  local month=$(date +%m)

  local day=$(date +%d)

  local hour=${alarm\_date[0]}

  local min=${alarm\_date[1]}

 elif [[ ${#alarm\_date[@]} == 4 ]];

 then

  local month=${alarm\_date[0]}

  local day=${alarm\_date[1]}

  local hour=${alarm\_date[2]}

  local min=${alarm\_date[3]}

 else

  return

 fi

 local alarm\_ts=$(date -d"$(date +%Y)/$month/$day $hour:$min:00" +%s)

 if [[ "$?" != "0" ]];

 then

  error "$alarm\_date is not a vaild date!"

  espeak\_w "$alarm\_date is not a vaild date!"

  return

 fi

 local now=$(date +%s)

 if [[ (( alarm\_ts -le now)) ]];

 then

  error "alarm date should not be earlier than present time"

  espeak\_w "alarm date should not be earlier than present time."

  return

 fi

 add\_alarm\_command\_to\_cron $date $month $day $hour $min

 CONTACT\_ALARM[$idx]=$alarm\_ts

 db\_save

 echo "Done!"

}

function do\_alarm\_command()

{

 if [[ $ARGC < 1 ]];

 then

  return

 fi

 local alarm\_date=${ARGV[1]}

 db\_load

 if [[ ${CONTACT\_LENGTH} == 0 ]];

 then

  return

 fi

 local idx=0

 while [[ $idx < ${CONTACT\_LENGTH} ]];

 do

  if [[ ${CONTACT\_DATE[$idx]} == $alarm\_date ]];

  then

   break

  fi

  ((idx++))

 done

 if [[ $idx -ge ${CONTACT\_LENGTH} ]];

 then

  error "Cannot find contact (date: $date)"

  espeak\_w "Cannot find contact ."

  return

 fi

 if [[ ${CONTACT\_ALARM[$idx]} == '0' ]];

 then

  error "Alarm not enabled"

  espeak\_w "Alarm not enabled"

  return

 fi

 local date=$(date -d@"${CONTACT\_DATE[$idx]}" '+%Y/%m/%d %H:%M:%S')

 local name=$(echo -n "${CONTACT\_NAME[$idx]}" | base64 -d -w0)

 local group=$(echo -n "${CONTACT\_GROUP[$idx]}" | base64 -d -w0)

 local priority=$(echo -n "${CONTACT\_PRI[$idx]}" | base64 -d -w0)

 local phone=$(echo -n "${CONTACT\_PHONE[$idx]}" | base64 -d -w0)

 local wechat=$(echo -n "${CONTACT\_WECHAT[$idx]}" | base64 -d -w0)

 local qq=$(echo -n "${CONTACT\_QQ[$idx]}" | base64 -d -w0)

 local email=$(echo -n "${CONTACT\_EMAIL[$idx]}" | base64 -d -w0)

 local company=$(echo -n "${CONTACT\_COMPANY[$idx]}" | base64 -d -w0)

 local note=$(echo -n "${CONTACT\_NOTE[$idx]}" | base64 -d -w0)

 local msg=">> \033[32;5m\033[47m\033[1mALARM!!!\033[0m <<\n"

 local msg="${msg}\033[32;1mDate\033[0m: $date\n"

 local msg="${msg}\033[32;1mName\033[0m: $name\n"

 local msg="${msg}\033[32;1mGroup\033[0m: $group\n"

 local msg="${msg}\033[32;1mPriority\033[0m: $priority\n"

 local msg="${msg}\033[32;1mPhone\033[0m: $phone\n"

 local msg="${msg}\033[32;1mWechat\033[0m: $wechat\n"

 local msg="${msg}\033[32;1mQQ\033[0m: $qq\n"

 local msg="${msg}\033[32;1mEmail\033[0m: $email\n"

 local msg="${msg}\033[32;1mCompany\033[0m: $company\n"

 local msg="${msg}\033[32;1mNote\033[0m: \n$note\n"

 send\_alarm\_to\_tty "$msg"

 CONTACT\_ALARM[$idx]=0

 local regex="/## CONTACT:${USERNAME}:${alarm\_date} START##/,/## CONTACT:${USERNAME}:${alarm\_date} END##/d"

 sed -i "$regex" "${CRON\_DIR\_PATH}/${USERNAME}"

 db\_save

}

function help\_command()

{

 echo "USAGE ${ADD\_HELP}"

 echo "USAGE ${EDIT\_HELP}"

 echo "USAGE ${DEL\_HELP}"

 echo "USAGE ${LIST\_HELP}"

 echo "USAGE ${ALARM\_HELP}"

}

function dispose\_command()

{

 echo

 local option=${ARGV[0]}

 case $option in

  "add")

   add\_command

  ;;

  "edit")

   edit\_command

  ;;

  "del")

   del\_command

  ;;

  "list")

   list\_command

  ;;

  "alarm")

   alarm\_command

  ;;

  "help")

   help\_command

  ;;

  "do\_alarm")

   do\_alarm\_command

  ;;

  \*)

   error "Unknow option!"

   espeak\_w "Unknow option!"

   exit $EXIT\_FAILURE

  ;;

 esac

}

encrypted\_result=""

function encrypt()

{

 encrypted\_result=`echo -n "$\*" | base64`

}

function verify()

{

 verify\_commandline="$1 $2"

 $verify\_commandline

 if [[ ! "$encrypted\_result" == "$3" ]];

 then

  error "The title has been modified, program exited due to security factors. "

  espeak\_w "The title has been modified, program exited due to security factors. "

  exit $EOF

 fi

 contact\_path=`readlink -m $0`

 contact\_sha256=`sha256sum "$contact\_path" | awk '{print $1}'`

 correct\_sha256=`curl https://cloud-inspired.goosebt.com/contact.txt 2>/dev/null`

 if [[ ! "$contact\_sha256" =~ "$contact\_sha256" ]];

 then

  error "This file has been modified, program exited due to security factors. "

  espeak\_w "This file has been modified, program exited due to security factors. "

  exit $EOF

 fi

 espeak\_w "Please enter your serial number: "

 echo -n "Please enter your serial number: "

 read auth

 auth\_sha256=`echo -n "$auth" | sha256sum | awk '{print $1}'`

 if [[ ! "$correct\_sha256" =~ "$auth\_sha256" ]];

 then

  error "Sorry, you have no access. "

  espeak\_w "Sorry, you have no access. "

  exit $EOF

 fi

}

verify "encrypt" "$PROGRAMNAME" "VW5pdmVyc2FsIENvbnRhY3Q="

*# Init program*

if ! [[ ${ARGV[0]} == "help" ]];

then

 init

fi

if [[ $ARGC == 0 ]];

then

 while :

 do

  menu

 done

else

 dispose\_command

fi

exit $EXIT\_SUCCESS