NCID - Network Caller ID User Manual

Authors:
John L Chmielewski

Last edited: Apr 10, 2024

Table of Contents

Introduction

NCID (Network Caller ID) is Caller ID (CID) distributed over a network to a variety of devices and computers.

The NCID server monitors either a modem, Caller ID device, or gateway (e.g., SIP, VoIP, smartphones) for the CID data. The data is collected and sent, via TCP, to one or more connected clients. The server supports multiple gateways which can be used with or without a modem or device. The server also supports one line text messages.

The NCID project website is the central place to go for the latest downloads, updated documentation and user/technical support for the official NCID package and its optional client packages.

This document contains information on how to get started with NCID, the hardware needed, a Frequently Asked Questions (FAQ) section and a TODO list. It also provides information on supported clients, gateways and optional server features. Troubleshooting information is also provided.

Getting Started

Table of Contents

NCID can be overwhelming for users who have never used it. Current users of NCID are probably not aware of all of its features, or how to use them properly. This document will try to help with those cases. The FAQ should also be of some help.

In this document:

- *NCID is the package name*

> - *ncidd is the server name*
> - *ncid is the client name*
> - *Unix is a generic term to mean any UNIX-like or Linux-like operating system, e.g., Fedora, FreeBSD, Mac OS X, Debian, etc.*
> - *Uncommenting a line means to remove the **#***

Getting Started Index

> - *Install the NCID package*
> - *Configure the ncidd server*
> - *Configure the ncid client*
> - *Configure call log rotation*
> - *Configure yearly call log*
> - *NCID startup*

Install the NCID package

> *See the list of Package Distributions.*

> *SourceForge distributes NCID packages for several operating systems. In addition, operating system specific third parties distribute NCID. For example, there are Ubuntu and Fedora repositories that include NCID, making it easy to install with their package management applications; however, they are not always up-to-date with the latest version at SourceForge.*

> *The first step is to download and install the NCID server, optional gateways and client. You can download the NCID packages from sourceforge or a repository for your operating system if it is up to date.*

> *If you are using Fedora, Ubuntu, or raspios, be aware that NCID is split into multiple rpm and deb packages. The server package is required. The gateways package is needed if you are using a gateway instead of a modem. The client and default output modules package is needed if you want to use the basic ncid client instead of another supported client, or if you want to use an output module.*

> *The Macintosh shell package and the FreeBSD shell package on sourceforge have the complete NCID system.*

> *There are many Linux distributions based on Redhat or Ubuntu so the Fedora or Ubuntu packages may install just fine. (For example, Linux Mint is an operating system based on Ubuntu.) Refer to the **INSTALL-"operating system"** section for your "operating system".*

> *If you cannot locate a package, you can download the source, compile and install it. Refer to the INSTALL (generic) section of this documentation.*

Configure the ncidd server

> *Now that you have installed NCID, you need to set the country code to format the telephone number and configure the method used to obtain Caller ID:*

> - *set the country code to format the telephone number*
> - *a Caller ID modem connected to a Unix computer*
> - *the serial NetCallerID device connected to a Unix computer*
> - *the CTI Comet USB device via the NCID XDMF gateway*
> - *the Holtek HT9032D based PSTN Caller ID module via the NCID XDMF gateway*
> - *a Caller ID modem on a Windows computer via the NCID YAC (Yet Another Caller ID) gateway*
> - *VoIP (Voice over Internet Protocol) phones via the NCID SIP (Session Initiation Protocol) gateway*
> - *Whozz Calling (WC) Ethernet Link network device via the NCID WC gateway*

- *an Android smartphone via the NCID Remote Notifier (RN) gateway*

Set the country code to format the telephone number

You need to set the country code for your country from the country code list. Edit ***ncid.conf*** *and uncomment one of the following lines. No need to uncomment the default but you need to replace* ***<CountryCode>*** *with your country code:*

```
# set country = "US"             # Default: country is United States
# set country = "ZZ"             # No country code for country
# set country = "<CountryCode>"  # use your country code
```

Using a modem connected to a Unix computer

Most modern modems support Caller ID but to determine if yours does follow the steps below and then refer to Modem Caller ID Test.

The server needs to know which port the modem is on. The default port is different depending on the Operating System:

```
Unknown Operating System distribution:    /dev/modem
Fedora, Redhat, Debian, raspios, Ubuntu: /dev/ttyACM0
FreeBSD:                                  /dev/cuaU0
Mac OS X:                                 /dev/cu.usbmodem24680241
Cygwin:                                   /dev/com1
```

If the default modem port is incorrect, you need to enter the correct port by editing ***ncidd.conf*** *and either uncomment one of these lines, or add a line with the correct port:*

```
# set ttyport = /dev/cu.modem # default Mac OS X internal modem
# set ttyport = /dev/cu.usbmodem24680241 # Mac OS X USB modem
# set ttyport = /dev/ttyS0 # Linux Serial Port 0
# set ttyport = /dev/ttyACM0 # Linux USB modem 0
```

If you wish to use the internal hangup feature, you need to uncomment one of these lines if not using the default:

```
# set hangup = 0  # Default: do not terminate a call
# set hangup = 1  # terminate the call
# set hangup = 2  # generate FAX tones, then terminate the call
# set hangup = 3  # play an announcement then terminate the call
```

If you wish to use the external hangup feature, you need to uncomment one of these lines if not using the default:

```
# hupmode = 0     # Default: do not execute the hangup extension
# hupmode = 1     # terminate the call
# hupmode = 2     # generate FAX tones, then terminate the call
# hupmode = 3     # play an announcement then terminate the call
```

After modifying ***ncidd.conf****, start ncidd. If ncidd is already running, it must be restarted.*

> ***Using a serial NetCallerID device connected to a Unix computer***

> *The serial NetCallerID (on the Wayback Machine) device is no longer manufactured by Ugotcall, but you can sometimes find it on eBay.*

> *Even though it supports Caller ID, the NetCallerID device is not a modem and it does not support the NCID hangup feature. Start by hooking it up to a serial port and making the changes above. Once that is completed, make these additional changes to* ***ncidd.conf****:*

> *Uncomment this line:*

```
# set cidinput = 1
```

> *Leave all hangup and hupmode set lines commented, for example:*

```
# set hangup = 1
```

> *After modifying* ***ncidd.conf****, start ncidd. If ncidd is already running, it must be restarted.*

> ***Using the XDMF gateway***

> *You would use the xdmf2ncid gateway if you are using the CTI Comet USB device or the Holtek HT9032D based PSTN Caller ID module connected to a Linux based machine.*

> *We have a complete section devoted to the XDMF gateway. See xdmf2ncid setup in the Gateways topic for more information.*

> ***Using a Caller ID modem on a Windows computer via the NCID YAC gateway***

> *You would use the YAC gateway if you are using a YAC server on a Windows computer running Microsoft Windows 98 or later.*

> *We have a complete section devoted to the YAC gateway. See yac2ncid setup in the Gateways topic for more information.*

> ***Using the SIP Gateway***

> *You would use the sip2ncid gateway if your VoIP phone is using SIP.*

> *We have a complete section devoted to the SIP gateway. See sip2ncid setup in the Gateways topic for more information.*

> ***Using the Whozz Calling (WC) gateway***

> *You would use the wc2ncid gateway if you are using a Whozz Calling Ethernet Link device. Serial Whozz Calling units are not currently supported.*

> *We have a complete section devoted to the WC gateway. See wc2ncid setup in the Gateways topic for more information.*

> ***Using the Android smartphone Remote Notifier (RN) gateway***

> *You would use the rn2ncid gateway if you are using an Android smartphone.*

> *We have a complete section devoted to the RN gateway. See rn2ncid setup in the Gateways topic document for more information.*

Configure the ncid client

> *Normally the client does not need to be configured, but you should review **ncid.conf** to see if you want to change the defaults for displaying the number format, displaying the date format and ring options if using an output module. There are other changes you can also make.*

> *After making any changes to **ncid.conf**, start the ncid client.*

> ***IMPORTANT:** The ncidd server and any needed gateways should already be running.*

Configure call log rotation

> *The /etc/ncid/ncidrotate.conf file controls how or if the call log is rotated:*

- *Set **RotateOff=0** and **Lines2keep=0** to backup and then empty the call log each month. This is the default and is required for the yearly call log.*
- *Set **RotateOff=1** to let the call log keep growing until the operating system decides it is too large.*
- *Set **Lines2keep=<NUMBER>** to backup and then keep **NUMBER** of lines in the call log at the start of the month.*

> *The /etc/ncid/rotatebysize.conf file allows the call log to grow to a minimum size before it is rotated:*

- *Set **minsize=1** to rotate the call log monthly. This is the default and is required for the yearly call log.*
- *Set **minsize=<NUMBER>** to grow the call log to **NUMBER** before rotation.*

> *Review Log Files for more information on log files.*

Configure yearly call log

> *The yearly crontab file is built each month by **ncid-yearlog**. It is designed to be run by crontab on the first of each month. It creates the $HOME/ncid/log directory the first time it is run. On the first of any month, it creates*

```
    $HOME/ncid/log/cidcall-<year>
```

> *and continues to add months until the end of the year.*

> *You need to add the crontab line in **REQUIREMENTS** by creating or modifying a crontab file using:*

> *crontab -e*

> ***IMPORTANT:***

- *The **ncid-yearlog** script should only be run once on the first of any month.*
- *If it is run on any day other than the first, it will do nothing.*

> ***REQUIREMENTS:***

- *logrotate installed and used by the operating system*

- *ncidrotate.conf must have **Lines2keep=0** (default)*

- *rotatebysize.conf must have **minsize 1** (default)*

- *user crontab:*

```
11 5 1 * * test -x bin/ncid-yearlog && bin/ncid-yearlog
```

NCID Startup

At this point you should have NCID functional. The INSTALL section for your operating system explains how to make sure NCID is working properly.

The INSTALL section also explains how to start NCID at boot and how to manually start the server, gateways and client.

If you are having any problems you can ask for assistance in the NCID Help or Open Discussion forums on SourceForge.

Installation

Table of Contents

Description

Description

NCID supports the operating systems indicated here. These are install instructions for each of the operating systems. There is also a generic install with compile instructions.

INSTALL (generic)
INSTALL-DEB (Debian, raspios or Ubuntu)
INSTALL-Cygwin
INSTALL-Fedora
INSTALL-FreeBSD
INSTALL-Mac
INSTALL-Redhat
INSTALL-Win

Review the INSTALL for your Operating System.

Generic INSTALL and Overview

If you're using a gateway, review the appropriate section in Gateways.

Table of Contents

Sections

- *LAYOUT:*
- *COMPILE:*
- *INSTALL:*
- *TEST USING a Modem:*

- *TEST USING a Device (like the NetCallerID box):*
- *TEST USING a Gateway:*

LAYOUT:

Programs, config files, modems, devices and log files

- *The programs go into* `$prefix/bin` *and* `$prefix/sbin` .
- *The config file goes into* `$prefix2/etc` .
- *The modem device is expected in* `$prefix3/dev` .
- *The LOG file is expected in* `$prefix3/var/log` .
- *The man pages go into* `$MAN` *which is* `$prefix/share/man` .
- *The Makefile targets are determined by the defaults used*
- *To view the target list with the prefix settings, type: make usage*

OS that uses systemd service files during boot

- *The service files go into $prefix/usr/lib/systemd/system .*

RPM based OS (Fedora, Redhat and CentOS) that uses init files during boot

- The init scripts go into `$prefix2/etc/rc.d/init.d` .

Debian based OS (Debian, raspios on Raspberry Pi and Ubuntu) that uses init files during boot

- The init scripts go into `$prefix2/etc/init.d` .

FreeBSD uses rc files during boot

- The rc scripts go into `$prefix2/etc/rc.d` .

Macintosh OSX

- The plist scripts go into `$(prefix3)/Library/LaunchDaemons` .

COMPILE:

Requirements

- *gmake (GNU make)*
- *gcc, g++ and c++ compilers*
- *libpcap and header files are needed for*
  - `sip2ncid`

- *libpcre2 and header files are needed for*
  - `cidupdate`
  - `ncidd`
  - `ncidnumberinfo`

- *libhidapi and header files are needed for*

    - `artech2ncid`
    - `cideasy2ncid`

- *libphonenumber and header files are needed for*
    - `cidupdate`
    - `ncidd`
    - `ncidnumberinfo`

- *protobuf and header files are needed for*
    - `cidupdate`
    - `ncidd`
    - `ncidnumberinfo`

- *libicu and header files are needed for*
    - `cidupdate`
    - `ncidd`
    - `ncidnumberinfo`

- *python3-pytz, python3-phonenumbers, and python3-dialog are needed for*
    - `phonetz` , `us_number_info` , `message_dialog` *- ncid client plugins*
- *python3-phonenumbers is needed for*
    - `us_number_info` *- used by ncid*

- *tcl is needed for*
    - `ncid` *non-graphical client and all output modules*

- *tk is needed for*
    - `ncid` *GUI client*

- *bwidget is needed for*
    - `ncid` *GUI client*

- *Perl module Config::Simple is needed for*
    - `email2ncid.pl`
    - `obi2ncid`
    - `rn2ncid`
    - `wc2ncid`
    - `wct`

    - `xdmf2ncid`

- *Perl module Data::HexDump is needed for*
    - `wc2ncid`
    - `wct`
    - `xdmf2ncid`

Notes

- *The Makefiles require GNU make.*
- *All of the supported OS distributions should have `libpcap` in their repositories. If it's missing, obtain it from TCPDUMP & LIBPCAP.*
- *Each supported OS distribution can have a different package manager, and the package names can also be different. See Installation and look for an INSTALL-<name> appropriate for your OS.*
- *For the required Perl modules, if they are available for installation using the OS distribution package manager, do so. This results in the best experience in terms of updates and security fixes. However, if they are not available, they can be installed using Perl's native package manager called CPAN. There are a few different ways that CPAN might have been configured as to how it handles root or sudo access, and generally it is safe to assume it was configured for the simplest usage.*

If you only need to install one of the packages:

```
    cpan Config::Simple
```

or

```
    cpan Data::HexDump
```

You can combine them on one line if you need both:

```
    cpan Config::Simple Data::HexDump
```

Compile to your desired directory structure

See the top of the Makefile for more information on targets.

- *To compile programs and config files for /usr/local:*

```
    make local
```

- *To compile programs for /usr and the config file for /etc:*

```
    make package
```

INSTALL:

> *See the top of the Makefile for more information on targets.*

- *To install in `/usr/local` (man pages go into `/usr/local/share/man`):*

```
sudo make install
```

- *To install in `/usr/local` (man pages go into `/usr/local/man`):*

```
sudo make install MAN=/usr/local/man
```

- *To install programs in `/usr`: config file in `/etc`,*
  and man pages in `/usr/share/man`:

```
sudo make package-install
```

or

```
sudo make install prefix=/usr prefix2=
```

TEST USING a Modem:

- *Start in this order (you may need `sudo ncidd` to access the modem):*

```
ncidd
ncid
```

- *Call yourself.*

- *If you have problems, start `ncidd` in debug mode:*

```
ncidd -D
```

- *To get more information, add the verbose flag:*

```
ncidd -Dv3
```

- *To also look at the alias, blacklist and whitelist structure:*

```
ncidd -Dv9
```

- *The last three lines will be similar to:*

```
Network Port: 3333
Wrote pid 20996 in pidfile: /var/run/ncidd.pid
End of startup: 04/01/2016 20:28:06
```

- *If `ncidd` aborts when you call yourself with something like:*

  ```
  Modem set for CallerID.
  Modem Error Condition. (Phone rang here)
  /dev/ttyS1: No such file or directory
  ```

  You need to set `ncidd` to ignore modem signals.

  Uncomment the following line in `ncidd.conf` :

  ```
        # set ttyclocal = 1
  ```

- *You should see the Caller ID lines between the first and second RING.*

- *If Caller ID is not received from the modem and if gencid is not set you will only see RING for each ring.*

- *If gencid is set (the default), you will get a CID at RING number 2:*

  ```
  07/13/2010 15:21  RING No Caller ID
  ```

This indicates one of three problems:

- *The modem is not set for Caller ID.*
- *The modem does not support Caller ID.*
- *The Telco is not providing Caller ID.*

- *Once you solve the problems, restart `ncidd` normally.*

TEST USING a Device (like the NetCallerID box):

- *Start in this order:*

  ```
  ncidd
  ncid
  ```

- *Call yourself.*

- *If you have problems, start `ncidd` in debug mode:*

  ```
  ncidd -D
  ```

- *To get more information, add the verbose flag:*

  ```
  ncidd -Dv3
  ```

- *To also look at the alias, blacklist and whitelist structure:*

  ```
  ncidd -Dv9
  ```

- *The last three lines will be similar to:*

```
Network Port: 3333
Wrote pid 20996 in pidfile: /var/run/ncidd.pid
End of startup: 04/01/2016 20:28:06
```

- *Once you solve any problems, restart* `ncidd` *normally.*

TEST USING a Gateway:

- *You may need to configure options. Review the appropriate section in Gateways.*

- *Start in this order:*

```
ncidd
<name of gateway>
ncid
```

For example:

```
ncidd
sip2ncid
ncid
```

- *Call yourself.*

- *If you have problems, start* `ncidd` *in debug mode:*

```
ncidd -D
```

- *To get more information, add the verbose flag:*

```
ncidd -Dv3
```

- *To also look at the alias, blacklist and whitelist structure:*

```
ncidd -Dv9
```

- *The last three lines will be similar to:*

```
Network Port: 3333
Wrote pid 20996 in pidfile: /var/run/ncidd.pid
End of startup: 04/01/2016 20:28:06
```

- *Once you solve any problems, restart* `ncidd` *normally.*

Cygwin Package Install

NCID version 1.12 and later do not support Cygwin.

NCID version 1.12 and later requires libphonenumber which is not part of the Cygwin distribution.

The Cygwin information below is provided as historical reference.

> *If NCID does not work, see INSTALL for some simple tests.*

> *If you're using a gateway, review the appropriate section in Gateways.*

> *Table of Contents*

Sections:

NOTES:

> *The NCID server cannot directly control a modem under Cygwin. Use the supplied yac2ncid gateway to control a modem under Windows.*

> *The sip2ncid gateway is not part of the NCID install package. It uses a network library not supported by Cygwin.*

> *The server must be configured in `ncidd.conf` for either:*

- *cidinput = 1 # Caller ID from a serial or USB device and optional gateways*
- *cidinput = 2 # Caller ID from gateways with modem support*
- *cidinput = 3 # Caller ID from gateways without modem support*

> *In a normal Unix installation, the ncid client script will automatically run in GUI mode under X-Windows. However, X-Windows is not available for Cygwin. Two ncid client options are available:*

- *Use the ncid client script in character mode under Cygwin:*

```
ncid --no-gui &
```

> *This also allows the use of output modules:*

```
ncid --no-gui <module> &
```

- *Download the ncid client Windows installer from SourceForge:*

```
ncid-VERSION-client_windows_setup.exe
```

This does run in GUI mode but does not allow the use of output modules.

INSTALL:

Install Cygwin from https://cygwin.com/

- *download* `setup-x86.exe`
- *run* `setup-x86.exe`
- *select cygwin download site (we used US site https://cygwin.osuosl.org)*
- *add the following to the default install setup*

  - *Devel/gcc-core*
  - *Devel gcc-g++*
  - *Devel/make*
  - *Editors/vim*
  - *Interpreters/perl*
  - *Interpreters/perl_pods*
  - *Interpreters/tcl*
  - *Net/openssh*
  - *text/pcre2*
  - *Libs/libpcre2-devel*
  - *Libs/hidapi-devel*

- *it is strongly recommended you enable cut and paste in the Cygwin window*

  - *Left click on the icon in upper left*
  - *Select Properties*
  - *Check QuickEdit Mode in Edit Options*

- *if compiling from source:*

  - *Download WinPcap Developer's Pack from https://www.winpcap.org/devel.htm*
  - *Unzip WpdPack_<version>.zip*
  - *Rename to WpdPack and move it to *

Install or upgrade NCID:

The NCID package normally installs in `/usr/local` *:*

- *Install or upgrade using the tar archive, if available:*

  - *Extracting the tar file will REPLACE the contents of all of the NCID configuration files. Be sure to back them up first. This includes all files in* `/usr/local/etc/ncid/` *.*

  - *Copy* `ncid-VERSION-cygwin.tgz` *to your Cygwin home directory.*

  - *Extract:*

    ```
    sudo tar -xzvf ncid-VERSION-cygwin.tgz -C /
    ```

Example:

```
sudo tar -xzvf ncid-1.12-cygwin.tgz -C /
```

- *Install or upgrade using the install script, if available.*

  - *For an upgrade, the install script will preserve existing configurations and new ones installed will have *.new as the extension.*

  - *Copy `ncid-VERSION-cygwin_install.sh` to your Cygwin home directory.*
  - *Run install script: sudo sh ncid-VERSION-cygwin_install.sh*

Example:

```
sudo sh ncid-1.12-cygwin_install.sh
```

- *If there is no binary package, you need to compile from source:*

  - *Copy `ncid-VERSION-src.tar.gz` to your Cygwin home directory.*

  - *Extract and compile. It will be installed to `/usr/local` (see top of Makefile):*

    ```
    tar -xzvf ncid-VERSION-src.tar.gz

    cd ncid

    make cygwin

    sudo make cygwin-install
    ```

If your phone system is VoIP and you want to use sip2ncid:

- *nothing else to do*

If you want to use a modem, you need YAC

- *download and install YAC (on the Wayback Machine)*
- *configure the YAC server for a listener at localhost (127.0.0.1)*

CONFIGURE:

The Makefile configures `ncidd.conf` for Cygwin, but you may want to change some of the defaults.

You need to configure sip2ncid to use the Network Interface. To find out the network interface name, you need to use the "-l" option to sip2ncid. You should see your Network interface names listed. Select the active one and use it with the "-i" option to sip2ncid.

START:

If this is your first time, you should first do the Test Using a Gateway and specify `sip2ncid` or `yac2ncid`.

> *Once testing is finished, run the processes in background:*

```
ncidd &
<name of gateway> &
ncid --no-gui &
```

> *Call yourself and see if it works.*

REBASE:

> *One of the idiosyncrasies of Cygwin is the need to rebase the dll's (set a base dll load address) so they don't conflict and create forking errors. The easiest way to do this is documented at Rebaseall.*

> *Just start an ash or dash prompt from `\\cygwin\\bin` and then type:*

```
rebaseall -v
exit
```

RUN AS A QUASI-SERVICE:

- *Don't do this process until you have ncidd and sip2ncid or other processes running properly. Once you have things setup though, you can set ncidd and sip2ncid to (sort of) run as a service in Windows. I only say "sort of" because it's not technically a service, but is called from another Cygwin component that is a service.*

- *Re-run the `setup.exe` that you used to install Cygwin and install the cygrunsrv package. It's under Admin.*

- *Go to a cygwin command line and type the following to install ncidd as a service:*

```
cygrunsrv -I ncidd -n -p /usr/local/bin/ncidd \
          -f "Network CallerID daemon(ncidd)" -a -D
```

> *Explaining these parameters:*

```
-I indicates install

-n indicates that the service never exits by itself (I don't
   recall why this has to be set, but it doesn't work otherwise)

-p /usr/local/bin/ncidd:
   Application path which is run as a service.

-f "Network CallerID daemon (ncidd)":
   Optional string which contains  the service description
   (the desc you see in the Services listing)

-a -D: passes the parameter "-D" to the ncidd program so it
   runs in debug mode. This keeps ncidd running in the
   "foreground" of the  cygrunsrv process.
```

- *Likewise, to remove the ncidd service:*

```
cygrunsrv -R ncidd
```

- *To install sip2ncid to run in the background, the command line is similar:*

```
cygrunsrv -I sip2ncid -n -p /usr/local/bin/sip2ncid -y ncidd \
          -a '-i "/Device/NPF_{XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX}" \
          -D' -f "Service picks network SIP packets and sends to ncidd" \
          --termsig KILL
```

Explaining these parameters:

```
-I indicates install

-n indicates that the service never exits by itself (I don't
   recall why this has to be set, but it doesn't work otherwise)

-p /usr/local/bin/sip2ncid: Application path which is run as
   a service.

-y ncidd: adds a service dependency with the ncidd service so
   that the ncidd service gets started automatically when you
   start sip2ncid

-a '-i "/Device/NPF_{XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX}" -D':
   note the single and double quotes in this section. You need to
   replace XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX in the above
   with NETWORK_INTERFACE from way above. To be clear, you want to
   replace /Device/NPF_{XXXXXXXX-XXXX-XXXX-XXXX-XXXXXXXXXXXX}
   with NETWORK_INTERFACE from way above.

-f "Service to pick SIP packets from network and send to ncidd":
   Optional string which contains the service description
   (the desc you see in the Services listing)

--termsig KILL: termination signal to send. If you don't include
   this the service doesn't always get stopped.
```

- *Likewise, to remove the sip2ncid service:*

```
cygrunsrv -R sip2ncid
```

- *To install ncid-notify to run in the background, the command line is similar: cygrunsrv -I ncid-notify -p /bin/sh.exe -a *
  '-c "/usr/local/bin/ncid --no-gui --module ncid-notify"'
  -f "Service to use notify service to send ncid messages to iPad"

Explaining these parameters:

```
    -I indicates install

    -p /bin/sh.exe: Application path to run, which in this case is
       just sh.exe because ncid-notify is a shell script

    -a '-c "/usr/local/bin/ncid --no-gui  --module ncid-notify"'
        these are the parameters that get sent to sh.exe:

        -c "/usr/local/bin/ncid: this is the path to the ncid script

        --no-gui: tells ncid not to run as gui

        --module ncid-notify: tells ncid to pass data to "ncid-notify"

    -f "Service to use notify service to send ncid messages to iPad":
        Optional string which contains the service description
        (the desc you see in the Services listing)

    -y ncidd: you COULD also add this line to add a service dependency
       with the ncidd service so that the ncidd service gets started
       automatically when you start ncid-notify. I don't do this,
       because strictly speaking, you could be running ncidd on a
       different computer
```

- *Likewise, to remove the ncid-notify service:*

```
cygrunsrv -R ncid-notify
```

DEB Package Install for Debian, raspios and Ubuntu

> *May be valid for other DEB Operating Systems*

> *If NCID does not work, see INSTALL for some simple tests.*

> *If you're using a gateway, review the appropriate section in Gateways.*

Table of Contents

Sections:

COMPILE/INSTALL from Source:

- *It's very important to update the latest package info before continuing. Don't skip these two steps!*
  (NOTE: The apt command is preferred over apt-get and apt-cache.)

```
sudo apt update
sudo apt upgrade
```

If any packages were listed as "kept back" and "not upgraded", do:

```
sudo apt dist-upgrade
```

- *The following packages are required:*

```
sudo apt install build-essential fakeroot
sudo apt install libpcre2-dev libhidapi-dev
sudo apt install libphonenumber-dev libicu-dev
sudo apt install libpcap-dev zstd rename
sudo apt install libconfig-simple-perl libdata-hexdump-perl
```

- *The following python3 packages are required:*

sudo apt install python3-pytz python3-phonenumbers python3-dialog

- *These packages are required to run the ncid GUI:*
  (pkg tk also installs tcl)

sudo apt install tk bwidget

- *The Perl package required for the email2ncid, obi2ncid, rn2ncid, wc2ncid, wct, and xdmf2ncid gateways:*

sudo apt install libconfig-simple-perl

- *This Perl additional package is required to run wc2ncid, wct and xdmf2ncid:*

sudo apt install libdata-hexdumper-perl

- *If the above Perl packages are not in the repository, you can try installing with the native Perl package manager called **cpan**:*

```
cpan install Config::Simple
cpan install Data::HexDump
```

- *Download the source from SourceForge:*

```
wget https://sourceforge.net/projects/ncid\
       /files/ncid/<version>/ncid-<version>-src.tar.gz
```

Example:

```
wget https://sourceforge.net/projects/ncid\
       /files/ncid/1.12/ncid-1.12-src.tar.gz
```

- *Copy `ncid-<version>-src.tar.gz` to any convenient directory, then type the following, where <os> is **debian**, **raspbian** or **ubuntu**:*

```
tar -xzvf ncid-<version>-src.tar.gz

cd ncid

make <os>

sudo make <os>-install
```

INSTALL/UPGRADE from DEB Package:

NCID requires the server and client DEB packages to function. The server is required on one computer or device, but the client can be installed on as many computers as needed.

If available, the latest NCID can be installed from a Debian, raspios or Ubuntu repository using apt.

If you cannot find a repository that contains NCID, or if the latest packages are not available, you can download them from SourceForge and install them using gdebi or dpkg.

In the sections below:

- *<version> represents the NCID version number (e.g., 1.12)*
- *<arch> is the architecture*
  - *use <armhf> for raspios 32 bit processors*
  - *use <amd64> for 64 bit processors*
- *<module> would be a module name like: kpopup, mythtv, samba*

It's very important to update the latest package info before continuing. Don't skip these two steps! (NOTE: The apt command is preferred over apt-get and apt-cache.)

```
sudo apt update
sudo apt upgrade
```

- ***Install NCID from a repository***

- *List the available packages:*

```
sudo apt search ncid
```

- *Install the server package (required):*

```
sudo apt install ncid_<version>-1_<arch>.deb
```

- *Install the client package (optional; includes most of the output modules):*

```
sudo apt install ncid-client_<version>-1_all.deb
```

- *Install any additional module packages wanted (optional):*

```
sudo apt install ncid-<module>_<version>-1_all.deb
```

- *Install the gateways package if using a gateway instead of a modem (optional):*

```
sudo apt install ncid-gateways_<version>-1_<arch>.deb
```

- ***Install NCID from DEB packages at SourceForge***

If the latest packages are not available at a repository, download them SourceForge.

Assuming the latest version is 1.12 and you're installing for a 64 bit processor, you would do the following:

- *Download the server package (required)*

```
wget https://sourceforge.net/projects/ncid\
       /files/ncid/1.12/ncid_1.12-1_amd64.deb
```

- *If using the client, download ncid-client (optional; includes most of the output modules):*

```
wget https://sourceforge.net/projects/ncid\
       /files/ncid/1.12/ncid-client_1.12-1_all.deb
```

- *Download any additional output modules (optional):*

```
wget https://sourceforge.net/projects/ncid\
       /files/ncid/1.12/ncid-<module>_1.12-1_all.deb
```

- *If using a gateway instead of a modem, download ncid-gateways (optional):*

```
wget https://sourceforge.net/projects/ncid\
       /files/ncid/1.12/ncid-gateways_1.12-1_amd64.deb
```

- *You can use apt, gdebi or dpkg to install the downloaded NCID packages and dependent packages.*

  - *Method 1: Install or Upgrade the packages using apt*

    - *Install the server (required):*

```
sudo apt install ./ncid_<version>-1_<arch>.deb
(note: the dot-slash is required here)
```

    - *Install the client (optional; includes most of the output modules):*

```
sudo apt install  ./ncid-client_<version>-1_all.deb
```

    - *Install any additional modules (optional):*

```
sudo apt install ./ncid-<module>_<version>-1_all.deb
```

- *Install any gateways (optional):*

```
sudo apt install ./ncid-gateways_<version>-1_<arch>.deb
```

- *Method 2: Install or Upgrade the packages using gdebi-gtk (GUI):*
  - *If needed, use the menu item "Add/Remove..." to install the GDebi Package Installer.*
  - ```
    Using the file viewer:

- Open the file viewer to view the NCID DEB packages
- Select the DEB packages
- Double-click selections or right-click selections and select
```

*"Open with GDebi Package installer"*

- *Method 3: Install or Upgrade the packages using gdebi (command line):*
  - *Install gdebi if needed:*

```
sudo apt install gdebi
```

- *Install the server (required):*

```
sudo gdebi ncid-<version>-1_<arch>.deb
```

- *Install the client (optional; includes most of the output modules):*

```
sudo gdebi ncid-client-<version>-1_all.deb
```

- *Install any additional modules (optional):*

```
sudo gdebi ncid-<module>-<version>-1_all.deb
```

- *Install any gateways (optional):*

```
sudo gdebi ncid-gateways_<version>-1_<arch>.deb
```

- *Method 4: Install or Upgrade the packages using dpkg (command line):*
  - *Install the server (required):*

```
sudo dpkg -i ncid-<version>_<arch>.deb
```

- *Install the client (optional; includes most of the output modules):*

```
sudo dpkg -i ncid-client-<version>-1_all.deb
```

- *Install any additional modules (optional):*
```

```
sudo dpkg -i ncid-<module>-<version>-1_all.deb
```

- *Install any gateways (optional):*

```
sudo dpkg -i ncid-gateways_<version>-1_<arch>.deb
```

- *Force install of all dependencies:*

```
sudo apt-get install -f
```

CONFIGURE:

> *The ncidd.conf file is used to configure ncidd. A typical installation places the ncid configuration files in /etc/ncid/.*

- *The default modem port in ncidd is /dev/ttyACM0. If you need to change it, set your modem port in ncidd.conf. This assumes serial port 0:*

```
   set ttyport = /dev/ttyS0
```

- *If you are using a Gateway to get the Caller ID instead of a local modem, you need to set cidinput:*

```
   set cidinput = 1 # Caller ID from a serial or USB device and optional
gateways
   set cidinput = 2 # Caller ID from gateways with modem support
   set cidinput = 3 # Caller ID from gateways without modem support
```

- *If you are using a local modem with or without a Gateway:*

```
   set cidinput = 0  (this is the default)
```

FIRST STARTUP:

- *If you are running the server and client on the same computer and using a modem:*

```
   sudo systemctl start ncidd

   ncid &
```

- *If you are running the server and using a gateway:*

```
   sudo systemctl start ncidd <name of gateway>

   ncid &
```

- *Call yourself and see if it works, if not:*

> *stop the server and gateway:*

```
    sudo systemctl stop mcid wc2ncid
```

> *and continue reading the test sections.*

- *If everything is OK, enable the NCID server, gateways and client modules you are using to autostart at boot. There are rc.init scripts for starting ncid with output modules, for example: ncid-page, ncid-kpopup, etc.*

> ***NOTE:***
>
> *The ncid GUI client must be started after login, not boot. There is no ncid.init script to start or stop it.*

START/STOP/RESTART/RELOAD/STATUS:

> *Use the systemctl command to control any of the daemons. The systemctl commands are: start, stop, restart, reload and status. The client using an output module can also be started using the* ***ncid-<module>*** *instead of* ***ncid****. All output modules can be run at the same time.*

> *Here are some examples:*

- *Start the NCID server:*

```
    sudo systemctl start ncidd
```

- *Stop the sip2ncid server:*

```
    sudo systemctl stop sip2ncid
```

- *Restart the sip2ncid server:*

```
    sudo systemctl restart sip2ncid
```

- *Reload the server alias file:*

```
    sudo systemctl reload ncidd
```

- *Start ncid with ncid-page:*

```
    sudo systemctl start ncid-page
```

- *Status of ncid with ncid-speak:*

```
    sudo systemctl status ncid-speak
```

> *Review the man page:* ***man systemctl***

AUTOSTART:

> *Use the systemctl command to enable/disable the service at boot.*

> *Here are some examples:*

- *Start ncidd at boot:*

```
sudo systemctl enable ncidd
```

- *Start ncid-page at boot:*

```
sudo systemctl enable ncid-page
```

- *Disable ncidd startup at boot:*

```
sudo systemctl disable ncidd
```

> *Review the man page:* ***man systemctl***

> *See also [this section about a known issue where ModemManager may hang NCID at boot time](#).*

LIST PACKAGE FILES:

- *To see all the files the package installed onto your system:*

```
dpkg-query -L <package_name>
```

- *To see the files a .deb file will install*

```
dpkg-deb -c <package_name.deb>
```

- *To work directly with package names rather than package files, you can use apt-file.*
  It lists contents for packages in your already-configured Apt repositories.
  It is irrelevant whether any particular package is or is not installed.

> *You may need to install the apt-file package first:*

```
sudo apt-file update
apt-file list package_name
```

> *After installing apt-file:*

```
apt-file list <package_name>
```

PACKAGE REMOVAL:

> *Use apt to remove any NCID package installed.*

> *For example, to use apt to remove the ncid package:*

- *Normal removal without removing configuration files and dependencies:*

```
sudo apt remove ncid
```

> - *Complete removal including configuration files:*

```
sudo apt purge ncid
```

> - *Remove ncid dependencies no longer needed:*

```
sudo apt autoremove
```

> *Review the man page:* ***man apt***

PACKAGE Dependencies:

> *Three ways to check package dependencies:*

> - *apt show <path_to_uninstalled_package>*
> - *apt-cache depends <Installed_package>*
> - *dpkg -I <path_to_uninstalled_package>*

KNOWN ISSUE - MODEMMANAGER MAY HANG NCID AT BOOT TIME:

> ***Configurations known to have the issue:***

> - *You are running Ubuntu 14.xx or later with the ModemManager installed and running.*
> - *You are running Debian 9.xx or later and the ModemManager is not installed and/or is not running.*
> - *You are running Ubuntu Mate on an Raspberry Pi model 3.*
> - *You are running Raspbian Jessie on an Raspberry Pi model 3.*
> - *It is a hardware modem mounted internally on a PCI card.*
> - *It is a USB modem.*

> ***Symptoms:***

> - *The NCID server is not sending caller ID to clients on your network.*
> - *Clients are unable to connect to the NCID server.*
> - *The NCID server log /var/log/ncidd.log indicates modem initialization did not complete and appears to be hung. This also prevents other processes after it from starting until the NCID server is killed.*
> - *You may see very strange modem responses in the NCID server log.*
> - *The NCID server problem is fixed by manually restarting it after the Operating System boots.*
> - *The NCID server problem is fixed by manually unplugging and re-plugging in the USB modem.*

> ***Solution: If the ModemManager is installed***

> *The ModemManager is searching for a mobile broadband (2G/3G/4G) capable devices, querying various Serial and USB connected devices at boot time, by sending "AT+GCAP" (the AT command for a modem to "Request Complete Capabilities List"). This can collide in terms of the devices initialization timing, with the NCID server and the XDMF to NCID gateway.*

> *The solution is to create the udev rules that will exclude from probing, devices used by the NCID server and XDMF to NCID gateway.*
> *Check out /usr/share/doc/ncid/README-udev for details.*

> *Disabling the ModemManager should be used as a last resort, if for some reason the method of the udev rules fails.*

> *Issue the following commands:*

```
    sudo systemctl disable ModemManager.service
    sudo systemctl stop ModemManager.service
    sudo systemctl status ModemManager.service
```

> *The **disable** line will prevent ModemManager from starting at boot.*

> *The **stop** line will terminate the currently running instance of ModemManager.*

> *The **status** lines should look like this:*

```
    ● ModemManager.service - Modem Manager
       Loaded: loaded (/usr/lib/systemd/system/ModemManager.service; disabled;
vendor preset: enabled)
       Active: inactive (dead)
```

> *In the cases where the Operating System has no systemctl available, disable ModemManager completely by removing its execute permissions, with the following command:*

```
    sudo chmod ugo-x /usr/sbin/ModemManager
```

Fedora RPM Package Install

> *If NCID does not work, see INSTALL for some simple tests.*

> *If you're using a gateway, review the appropriate section in Gateways.*

> *Table of Contents*

Sections:

COMPILE/INSTALL from Source:

> ***Compile using the ncid-<NCID version>-src.tar.gz tar archive:***

> *The following packages are required:*

```
    sudo dnf install libpcap-devel pcre2-devel hidapi-devel tcl
    sudo dnf install libphonenumber-devel libicu-devel
    sudo dnf install protobuf-devel
```

> *These packages are required to run the ncid GUI:*
> *(note that package tk also installs package tcl)*

```
    sudo dnf install tk bwidget
```

> *The following Python3 packages are required:*

```
sudo dnf install python3-pytz python3-phonenumbers python3-dialog
```

> *This Perl package is required to run email2ncid, obi2ncid, rn2ncid, wc2ncid, wct, and xdmf2ncid:*

```
sudo dnf install perl-Config-Simple
```

> *This additional Perl package is required to run wc2ncid, wct and xdmf2ncid:*

```
sudo dnf install perl-Data-HexDump
```

> *If the above Perl packages are not in the repository, you can try installing with the native Perl package manager called **cpan**:*

```
cpan install Config::Simple
cpan install Data::HexDump
```

> *Finally, compile and install:*

```
make fedora
sudo make fedora-install
```

> ***Rebuild the rpm packages using ncid-<NCID version>.fc<OS version>.src.rpm***

> *Install the required packages*

```
dnf builddep ncid-<NCID version>.fc<OS version>.src.rpm
```

> *Rebuild the RPM packages*

```
rpm --rebuild ncid-<NCID version>.fc<OS version>.src.rpm
```

INSTALL/UPGRADE from RPM Package:

> *NCID requires the server and client RPM packages to function. The server is required on one computer or device, but the client can be installed on as many computers as needed.*

> *The client has most of the output modules in its RPM package, but there are optional output modules in their own RPM packages.*

> *Download the server and client RPM packages using dnf from the Fedora repositories. You can also download any optional output modules you want.*

- *List the available packages:*

  ```
  sudo dnf list ncid\*
  ```

- *The most recent versions may be here:*

  ```
  sudo dnf install fedora-release-rawhide
  sudo dnf --enablerepo=rawhide list ncid\*
  ```

- *Install the server package (required):*

```
sudo dnf install ncid-< rpm package >
```

- *Install the client package (optional)*

```
sudo dnf install ncid-client-< rpm package >
```

- *Install the gateways package if using a gateway instead of a modem (optional):*

```
sudo dnf install ncid-gateways-< rpm package >
```

- *Install any optional module packages wanted (most modules are included with the client package):*

```
sudo dnf install ncid-< module rpm package >
```

If this is an upgrade and you changed any NCID configuration files, RPM creates new files with either .rpmnew (the new default config file), or .rpmsave (your old config file backed up). The rpmconf tool simplifies updating the configuration files from .rpmnew from .rpmsave.

- *Install rpmconf, if needed:*

```
sudo dnf install rpmconf
```

- *Update all configuration files that have .rpmnew or .rpmsave files:*

```
sudo rpmconf -a
```

If the current release is not in the Fedora repositories, download the RPM packages from https://sourceforge.net/projects/ncid/

- *Download server, gateways and client RPM Packages from SourceForge:*

```
ncid RPM Package          (server - required)
ncid-client RPM Package   (client & default output modules - optional)
ncid-gateways RPM Package (gateways - optional)
```

- *Download any optional output modules wanted from SourceForge:*

```
ncid-MODULE RPM Package  (optional client output modules)
```

- *Install or Upgrade the packages:*

  - *Using the file viewer:*

```
- Open the file viewer to view the NCID RPM packages
- Select the RPM packages
- Right click selections and select "Open with Package installer"
```

- ```
  Using dnf:
  ```

```
sudo dnf install ncid*.rpm
```

CONFIGURE:

The ncidd.conf file is used to configure ncidd.

- *The default modem port in ncidd is /dev/ACM0. If you need to change it, set your modem port in ncidd.conf. This assumes serial port 0:*

  ```
  set ttyport = /dev/ttyS0
  ```

- *If you are using a Gateway to get the Caller ID instead of a local modem, you need to set cidinput:*

  ```
  set cidinput = 1 # Caller ID from a serial or USB device and optional
gateways
  set cidinput = 2 # Caller ID from gateways with modem support
  set cidinput = 3 # Caller ID from gateways without modem support
  ```

- *If you are using a local modem with or without a Gateway:*

  ```
  set cidinput = 0  (this is the default)
  ```

FIRST STARTUP:

- *If you are running the server and client on the same computer and using a modem:*

  ```
  sudo systemctl start ncidd

  ncid &
  ```

- *If you are running the server and using a gateway:*

  ```
  sudo systemctl start ncidd <name of gateway>

  ncid &
  ```

- *Call yourself and see if it works, if not,*

stop the gateway and server:

```
sudo systemctl stop <name of gateway> ncidd
```
```

and continue reading the test sections.

- *If everything is OK, enable the NCID server, gateways and client modules you are using to autostart at boot.*

For example, to start ncidd and sip2ncid at boot:

```
sudo systemctl enable ncidd sip2ncid
```

The GUI ncid client must be started after login, not boot.

NOTE:

> *The ncid client normally starts in the GUI mode and there is no ncid.service script to start or stop it. There are service scripts for starting ncid with output modules, for example: ncid-page, ncid-kpopup, etc.*

> *The ncid client can be started automatically from the **Auto Start** item in the **File** menu.*

START/STOP/RESTART/RELOAD/STATUS:

Use the 'systemctl' command to control any of the daemons. The service commands are: start, stop, restart, reload, reload-or-restart and status. The client can also be started using the output module name instead of ncid. All output modules can be run at the same time.

Here are some examples:

- *Start the NCID server:*

```
sudo systemctl start ncidd.service
```

- *Stop the sip2ncid server:*

```
sudo systemctl stop sip2ncid.service
```

- *Reload the server alias file:*

```
sudo systemctl reload-or-restart ncidd.service
```

- *Restart ncid using ncid-page:*

```
sudo systemctl start ncid-page.service
```

- *Get the status of ncid using ncid-speak:*

```
sudo systemctl status ncid-speak.service
```

*Review the man page: **man systemctl***

AUTOSTART:

> *Use the 'systemctl' command to enable/disable a service to start at boot.*

> *Here are some examples:*

- *Autostart ncidd at boot:*

```
sudo systemctl enable ncidd
```

- *Autostart ncidd and sip2ncid at boot:*

```
sudo systemctl enable ncidd sip2ncid
```

- *Disable ncid-speak from starting at boot:*

```
sudo systemctl disable ncid-speak
```

> *Review the man page:* ***man systemctl***

FreeBSD Install

> *If NCID does not work, see INSTALL for some simple tests.*

> *If you're using a gateway, review the appropriate section in Gateways.*

> *Table of Contents*

Sections:

>

COMPILE:

> *See INSTALL.*

INSTALL:

> *The NCID package installs in /usr/local.*

> ***Requirements***

> *All packages can be installed from repositories or compiled from* ***ports****. Instructions assume the user* ***root*** *and use pkg to install from repositories. If the sudo command is installed, replace* ***pkg*** *with* ***sudo pkg****, if not root.*

- *update the repository data and upgrade:*

  pkg update
  pkg upgrade

- *Install sudo. The user needs to be in group **wheel**. A line in /usr/local/etc/sudoers needs to be uncommented to allow users in group wheel to run as root.*

  pkg install sudo

The NCID Makefile requires gmake and bash. To obtain the Makefile usage: gmake

pkg install bash gmake

- *X-windows and Gnome (NCID default GUI environment)*

  pkg install xorg urwfonts gnome3

add to /etc/rc.conf

dbus_enable="YES"
hald_enable="YES"
gdm_enable="YES"
gnome_enable="YES"

add to /etc/fstab (required by Gnome)

```
proc            /proc       procfs  rw  0   0
```

- *Compile ncid*

  pkg install libpcap pcre2 hidapi libphonenumber gmake gcc bwidget python3

- *install python39 py39-pytz py39-phonenumbers py39-dialog (or newer versions)*

  sudo pkg python39 py39-pytz py39-phonenumbers py39-dialog

- *install Perl config-simple and Data-HexDump*

  sudo pkg install p5-config-simple p5-Data-HexDump

- *install tk8.7 (tcl/tk version must be >= 8.6; pkg tk also installs tcl)*

  sudo pkg install tk87

- *make sure ncid calls tcl?.? or wish?.?*

  For example: tclsh8.7 and wish8.7 otherwise modify the TCLSH and WISH variables in ncid

- *install logrotate*

  pkg install logrotate

- *install the vim editor (or another editor of your choice)*

  pkg install vim

Install or upgrade using the install script as root, if available:

GNU getopt is required when using the install script. See this note.

*For an upgrade, the install script will preserve existing configurations and new ones installed will have *.new as the extension.*

```
Copy ncid-VERSION-freebsd_install.sh to the FreeBSD computer

sh ncid-VERSION-freebsd_install.sh
```

Example:

```
sh ncid-1.10-freebsd_install.sh
```

*You will need to manually compare your current configuration files with the *.new ones and manually edit any differences.*

If there is no binary package, you need to compile the source.

Gmake is required when compiling the source. See this note.

*Your existing configuration files will be preserved and new ones installed will have *.new as the extension.*

```
Copy ncid-VERSION-src.tar.gz to the FreeBSD computer

tar -xzvf ncid-VERSION-src.tar.gz

gmake freebsd (compiles for /usr/local, see top of Makefile)

gmake freebsd-install
```

CONFIGURE:

The ncidd.conf file is used to configure ncidd.

- *The default modem port in ncidd is a USB modem at /dev/cuaU0. There may also be /dev/ttyACM0 in addition to /dev/cuaU0.*

- *You should set the modem in ncidd.conf, if you need to change it. Use one of cuaa0, cuaa2, cuaa3, cuaa4 in /dev which corresponds to COM1, COM2, COM3, COM4*

If you are using COM1 then you would add this line to ncidd.conf:

```
 set ttyport = /dev/cuaa0
```

- *If you are using a gateway instead of a local modem, you need to set cidinput:*

```
        set cidinput = 1 # Caller ID from a serial or USB device and
optional gateways
        set cidinput = 2 # Caller ID from gateways with modem support
        set cidinput = 3 # Caller ID from gateways without modem support
```

- *If you are using a local modem with or without a gateway:*

```
        set cidinput = 0  (this is the default)
```

STARTUP:

- *If you are running the server and client on the same computer and using a modem:*

```
   /usr/local/etc/rc.d/ncidd onestart

   ncid &
```

- *If you are running the server and using a gateway:*

```
   /usr/local/etc/rc.d/rc.d/ncidd onestart

   /usr/local/etc/rc.d/rc.d/<name of gateway> onestart

   ncid &
```

- *Call yourself and see if it works, if not:*

stop the gateway used:

```
     /usr/local/etc/rc.d/rc.d/<name of gateway> onestop
```

stop the server:

```
     /usr/local/etc/rc.d/rc.d/ncidd onestop
```

and continue reading the test sections.

- *If everything is OK, enable the NCID server, gateways and client modules you are using to autostart at boot.*

- *NOTE:*

  - *The ncid client normally starts in the GUI mode and there is no ncid.rc script to start or stop it.*

  - *There are rc.d scripts for starting ncid with output modules, for example: ncid-page, ncid-kpopup, etc.*

START/STOP/STATUS:

> - *The /usr/local/etc/rc.d/ncid* scripts to control any of the daemons. The rc.d commands are: start, stop, restart, reload and status. The client can also be started using the output module name instead of ncid. All output modules can be run at the same time.*

> *Here are examples:*

```
sudo /usr/local/etc/rc.d/ncidd start
sudo /usr/local/etc/rc.d/ncidd reload
sudo /usr/local/etc/rc.d/sip2ncid restart
sudo /usr/local/etc/rc.d/ncid-speak stop
sudo /usr/local/etc/rc.d/ncid-page status
sudo /usr/local/etc/rc.d/ncid-kpopup rcvar
```

> - *If a service is not enabled, you must prefix 'one' to the commands; start becomes onestart, stop becomes onestop and so forth.*

AUTOSTART:

> - *If you want NCID services to start automatically at boot, you need to add an enable line /etc/rc.conf.local for each service you want started. If /etc/rc.conf.local does not exist, create it.*

> - *Here is the list of rc scripts and their enable lines:*

```
/usr/local/etc/rc.d/     /etc/rc.conf.local
--------------------     ------------------
ncidd                    ncidd_enable="YES"
sip2ncid                 sip2ncid_enable="YES"
yac2ncid                 yac2ncid_enable="YES"
ncid-kpopup              ncidkpopup_enable="YES"
ncid-notify              ncidnotify_enable="YES"
ncid-page                ncidpage_enable="YES"
ncid-samba               ncidsamba_enable="YES"
ncid-skel                ncidskel_enable="YES"
ncid-speak               ncidspeak_enable="YES"
ncid-yac                 ncidyack_enable="YES"
```

TRIMMING LOG FILES:

> - *FreeBSD uses newsyslog by default to trim files. To trim the cidcall.log and the ciddata.log files, add this entry to /etc/newsyslog.conf:*

```
/var/log/cid*.log   root:wheel 644 5 * $M1D0 GN
```

GMAKE NOTE:

> - *The NCID source package requires gmake.*

> - *The NCID source package and, if available, the install script, require GNU getopt. It is installed from /usr/ports/devel/libgnugetopt:*
>
>   ```
>   cd /usr/ports/devel/gmake
>
>   make all install
>   ```

Macintosh Install

> *If NCID does not work, see INSTALL for some simple tests.*

> *If you're using a gateway, review the appropriate section in Gateways.*

> *Table of Contents*

Sections:

SYSTEM REQUIREMENTS:

> ***macOS:***

> - *NCID should work on macOS 11.7 (Big Sur) and later versions. It requires an Intel 64bit processor.*

> ***Graphical User Interface (GUI):***

> - *The native macOS GUI called **Aqua** is not compatible with the NCID GUI client. Instead, it needs **XQuartz**, a macOS specific version of **X Windows** (a.k.a. **X11**). **XQuartz** is installed as a separate application and runs alongside **Aqua**.*

> - *In addition to **XQuartz**, a special version of TCL/TK is required that works with **XQuartz**. The steps to install these depend on which package manager is used and are incorporated below.*

> ***Package Manager:***

> *Pick one of the following package managers (do not install both).*

> ***Homebrew**:*
>
> - *This is the preferred package manager because it appears to be more actively maintained, especially for the latest versions of macOS. Check here to see if you meet the minimum system requirements.*
> - *Click here for instructions to download and install Homebrew.*

- *As of this writing for NCID 1.14, there is a bug in the Homebrew formula to install the special version of TCL/TK that is required to work with **XQuartz**: You cannot successfully install TCL/TK if the path to the Xcode Command Line Tools (CLT) contains an embedded space. This should only be an issue if you install the full Xcode application to a location other than the default, which is /Applications/Xcode. To determine if this issue applies to your macOS environment, examine the path emitted by the following command typed at a shell prompt:*

```
xcode-select -p
```

***MacPorts**:*

- *You will likely use MacPorts for older versions of macOS that Homebrew no longer supports.*
- *File **/etc/paths** needs to be edited to put in directories for finding MacPorts. Use **sudo** to make a backup first, then edit the file with:*

```
sudo vi /etc/paths
```

so that it looks like this:

```
/opt/local/libexec/gnubin
/opt/local/bin
/opt/local/sbin
/usr/bin
/bin
/usr/sbin
/sbin
/usr/local/bin
/usr/local/sbin
```

- *Click here for instructions to download and install MacPorts.*

- *A minimum of MacPorts version 2.8.0 is required. However, as of this writing for NCID 1.14, there is a bug in version 2.8.0 and later where the hidapi package will not install header file hidapi_darwin.h; this causes the compile to fail. This has been reported to the MacPorts maintainers.*

COMPILE/INSTALL/UPGRADE from Source:

Prerequisite:

- *When building NCID from source, you must compile on a case-sensitive macOS filesystem. This requirement is ONLY for compiling NCID. Do not attempt to put other macOS applications on a case-sensitive filesystem because they will probably not work as expected.*

- *Use the **diskutil** command to determine the filesystem type. Assuming you will be installing to the Mac's startup volume, i.e., the root filesystem, type the following:*

```
diskutil info / | fgrep -i "file system"
```

- *Depending on the version of macOS, the default filesystem type is 'APFS' or 'Journaled HFS+' which are not case-sensitive.*

- *Look for 'Case-sensitive' in the output of the **diskutil** command above. If you don't see it, you can create a small, case-sensitive disk image just so you can compile NCID. Approximately 100 megabytes are required for each version of NCID you intend to compile. A minimum 200 megabyte disk image size is recommended.*

```
hdiutil create -size 200m \
               -fs "Case-sensitive HFS+" \
               -volname NCID ~/NCID.dmg
```

- *Next, mount the disk image:*

```
hdiutil attach ~/NCID.dmg
```

- *and change to its directory where you will place the source:*

```
cd /Volumes/NCID
```

- *Rebooting the Mac computer will normally unmount the disk image requiring the **hdiutil attach** command to be manually run each time. You can set it up to automount by doing the following:*

  1. *Login as the user where you want to autmount NCID.dmg.*

  2. *From a Terminal prompt, mount the disk image:*
     hdiutil attach ~/NCID.dmg

  3. *There should now be an icon on the Desktop called NCID showing that it has been mounted.*

  4. *Go to System Preferences->Users & Groups->Login Items. Drag the NCID icon from the right edge of the Desktop and drop it onto the Login Items (top/bottom/doesn't matter where in the list).*

  5. *The Login Items window may not refresh automatically so click on the Password tab and then go back to the Login Items tab. You should then see NCID in the list of kind "Volume".*

Homebrew:

- *Do not use **sudo** when executing **brew install**.*

- *The following packages are required:*

```
brew tap sethrfore/homebrew-r-srf
brew install icu4c libpcap libphonenumber pcre2 hidapi
brew install make wget sethrfore/r-srf/tcl-tk-x11 bwidget
brew install xquartz
```

- *Download the source from SourceForge:*

```
wget https://sourceforge.net/projects/ncid\
       /files/ncid/<version>/ncid-<version>-src.tar.gz
```

Example:

```
wget https://sourceforge.net/projects/ncid\
       /files/ncid/1.14/ncid-1.14-src.tar.gz
```

- *Copy `ncid-<version>-src.tar.gz` to the macOS case-sensitive filesystem and then type:*

```
tar -xzvf ncid-<version>-src.tar.gz

cd ncid

make mac

sudo make mac-install
```

- *Change the default symbolic links for the TCL/TK interpreter to point to the **XQuartz** version by typing the following (this normally only needs to be done once or if Homebrew updates TCL/TK):*

```
ncid-setup mac-homebrew-tcl
```

- *Go to Applications->Utilities and launch **XQuartz**. This will create its default Preference file. It can take a minute for **XQuartz** to finish running the first time. This only needs to be done once.*

- *Configure **XQuartz** to run the NCID GUI client when **XQuartz** starts by typing the following (only needs to be done once):*

```
ncid-setup mac-xquartz-app-menu
```

MacPorts:

- *You must use **sudo** when executing **port install**.*

- *The following packages are required:*

```
sudo port install libpcap libphonenumber-cpp pcre2 gmake
sudo port install dos2unix wget bwidget hidapi abseil
sudo port install tk +quartz
```

- *Download the source from SourceForge:*

```
wget https://sourceforge.net/projects/ncid\
       /files/ncid/<version>/ncid-<version>-src.tar.gz
```

Example:

```
wget https://sourceforge.net/projects/ncid\
       /files/ncid/1.14/ncid-1.14-src.tar.gz
```

- *Copy `ncid-<version>-src.tar.gz` to the macOS case-sensitive filesystem and then type:*

```
tar -xzvf ncid-<version>-src.tar.gz

cd ncid

make mac

sudo make mac-install
```

- *Go to Applications->Utilities and launch **XQuartz**. This will create its default Preference file. It can take a minute for **XQuartz** to finish running the first time. This only needs to be done once.*

- *Configure **XQuartz** to run the NCID GUI client when **XQuartz** starts by typing the following (only needs to be done once):*

```
ncid-setup mac-xquartz-app-menu
```

Other Notes:

- *For both an install and an upgrade, existing configurations are automatically preserved, but new ones installed will have .new as the extension. You will need to manually compare your current configuration files with the ".new" ones and manually edit any differences.*

- *Optional gateways will require additional Perl packages before they can be run. Installing these with the native Perl package manager called **cpan**.*

  - *This package is required to run email2ncid, obi2ncid, rn2ncid, wc2ncid, wct, and xdmf2ncid:*

```
cpan install Config::Simple
```

  - *This additional package is required to run wc2ncid, wct and xdmf2ncid:*

```
cpan install Data::HexDump
```

CONFIGURE:

The Makefile preconfigures ncidd.conf for macOS, but you may want to change some of the defaults.

- *If you are using a gateway instead of a local modem, you need to set cidinput:*

```
set cidinput = 1 # Caller ID from a serial or USB device and optional gateways
set cidinput = 2 # Caller ID from gateways with modem support
set cidinput = 3 # Caller ID from gateways without modem support
```

- *If you are using a local modem with or without a gateway:*

```
    set cidinput = 0  (this is the default)
```

FIRST STARTUP:

> *NCID requires the server and at least one client to function. The server is required on one computer or device, but the client can be installed on as many computers as needed.*

- *If you are running the server and client on the same computer and using a modem:*

```
    sudo /usr/local/sbin/ncidd
```

> *In Finder, navigate to Applications->Utilities and double-click on **XQuartz** (or **XQuartz.app**).*

- *If you are running the server and using a gateway:*

```
    sudo /usr/local/sbin/ncidd

    sudo /usr/local/sbin/<name of gateway>
```

> *In Finder, navigate to Applications->Utilities and double-click on **XQuartz** (or **XQuartz.app**).*

- *Call yourself and see if it works. If not, stop the gateway first (if used) and then stop the server, using **sudo kill** and the appropriate process ID. Continue by reading the test sections.*

- *If everything is OK, enable the NCID server, gateways and client modules you are using to autostart at boot.*

(AUTO)START/STOP:

> ***SERVER:***

> *Under macOS the mechanism used to start the NCID server processes is **launchd** and requires **.plist** files in /Library/LaunchDaemons. The naming convention used is as follows:*

```
    /Library/LaunchDaemons/net.sourceforge.ncid-{name}.plist
```

> *Appropriate **.plist** files for the NCID server processes are created automatically when NCID is installed, however, they must be manually activated.*

> *Once activated, no action is typically required as the **.plist** files are configured to automatically start each time the system boots.*

> *You do not interact with **launchd** directly, instead you use the **launchctl** command line utility.*

> *You should only activate the NCID servers, gateways and client modules you need. Activating will also start the process immediately; there is no need to reboot.*

> *The syntax for stopping the daemons is the same as starting them, except you use the unload subcommand instead of the load subcommand. Doing an unload stops the daemon immediately and prevents it from starting automatically the next time the system is booted.*

> *Here are some examples:*

> - *Start the NCID server:*

```
sudo launchctl load -w \
     /Library/LaunchDaemons/net.sourceforge.ncidd.plist
```

> - *Stop the sip2ncid server:*

```
sudo launchctl unload -w \
     /Library/LaunchDaemons/net.sourceforge.sip2ncid.plist
```

> - *Start ncid with ncid-page:*

```
sudo launchctl load -w \
     /Library/LaunchDaemons/net.sourceforge.ncid-page.plist
```

> *Review the man page:* ***man launchctl***

> ***CLIENT:***

> *For the NCID GUI client, no* ***.plist*** *is currently provided because of the requirement that NCID must be installed as root and the GUI preference file is specific to each user.*

> *However, if you follow the steps described in this document,* ***XQuartz*** *will be configured to run the NCID GUI client when it launches. To have XQuartz launch when you automatically log in, drag* ***XQuartz*** *(or* ***XQuartz.app****) from Applications->Utilities to your account's Login Items: System Preferences->Users & Groups->Login Items tab.*

> *You may optionally want to drag the* ***XQuartz*** *icon from Applications->Utilities and put it in the Dock.*

CHECKING DAEMON STATUS:

> *Use the* ***launchctl*** *list subcommand to show the daemons currently loaded, optionally using* ***fgrep*** *to filter out only NCID related processes.*

> *Daemons currently running will have a process id.*

> *Daemons which were stopped without an error will not be listed at all.*

> *If a daemon has stopped due to an error, it will have no process id but will have a numeric exit status. Examine the contents of the /var/log/system.log file to determine the problem. Once you fix the problem, use the* ***launchctl*** *unload subcommand followed by the load subcommand.*

> *Example:*

```
sudo launchctl list|fgrep net.sourceforge.ncid

PID     Status  Label
422     -       net.sourceforge.ncid-notify
419     -       net.sourceforge.ncidd
```

TRIMMING LOG FILES:

> *macOS uses **newsyslog** to trim files. To trim the cidcall.log and the ciddata.log files, add this entry to /etc/newsyslog.conf*

```
    /var/log/cid*.log   root:wheel 644 5 * $M1D0 GN
```

Redhat/Centos/Enterprise RPM Package Install

> *If NCID does not work, see INSTALL for some simple tests.*

> *If you're using a gateway, review the appropriate section in Gateways.*

> *Table of Contents*

Sections:

COMPILE/INSTALL from Source:

> ***Compile using the ncid-<NCID version>-src.tar.gz tar archive:***

> > *The following packages are required:*

```
    sudo dnf install libpcap-devel pcre2-devel hidapi-devel tcl
    sudo dnf install libphonenumber-devel libicu-devel
    sudo dnf install protobuf-devel
```

> > *These packages are required to run the ncid GUI:*
> > *(note that package tk also installs package tcl)*

```
    sudo dnf install tk bwidget
```

> > *Python3 modules required by ncid client plugins:*

```
    sudo dnf install python3-pytz python3-phonenumbers python3-dialog
```

> > *This Perl package is required to run email2ncid, obi2ncid, rn2ncid, wc2ncid, wct, and xdmf2ncid:*

```
    sudo dnf install perl-Config-Simple
```

> > *This additional Perl package is required to run wc2ncid, wct and xdmf2ncid:*

```
    sudo dnf install perl-Data-HexDump
```

> > *If the above Perl packages are not in the repository, you can try installing with the native Perl package manager called **cpan**:*

```
    cpan install Config::Simple
    cpan install Data::HexDump
```

Finally, compile and install:

```
    make redhat>> This Python3 package is required to run us_number_info, used by
ncid:
```

```
    sudo dnf install python3-phonenumbers
```

```
    sudo make redhat-install
```

Rebuild the rpm packages using ncid-<NCID version>.fc<OS version>.src.rpm

Install the required packages

```
    dnf builddep ncid-<NCID version>.fc<OS version>.src.rpm
```

Rebuild the RPM packages

```
    rpm --rebuild ncid-<NCID version>.fc<OS version>.src.rpm
```

INSTALL/UPGRADE from RPM Package:

NCID requires the server and client RPM packages to function. The server is required on one computer or device, but the client can be installed on as many computers as needed.

The client has most of the output modules in its RPM package, but there are optional output modules in their own RPM packages.

Download the server and client RPM packages using dnf from the Fedora repositories. You can also download any optional output modules you want.

- *List the available packages:*

  ```
  sudo dnf list ncid\*
  ```

- *The most recent versions may be here:*

  ```
  sudo dnf install fedora-release-rawhide
  sudo dnf --enablerepo=rawhide list ncid\*
  ```

- *Install the server package (required):*

  ```
  sudo dnf install ncid-< rpm package >
  ```

- *Install the client package (optional)*

```
    sudo dnf install ncid-client-< rpm package >
```

- *Install the gateways package if using a gateway instead of a modem (optional):*

```
sudo dnf install ncid-gateways-< rpm package >
```

- *Install any optional module packages wanted (most modules are included with the client package):*

```
sudo dnf install ncid-< module rpm package >
```

If this is an upgrade and you changed any NCID configuration files, RPM creates new files with either .rpmnew (the new default config file), or .rpmsave (your old config file backed up). The rpmconf tool simplifies updating the configuration files from .rpmnew from .rpmsave.

- *Install rpmconf, if needed:*

```
sudo dnf install rpmconf
```

- *Update all configuration files that have .rpmnew or .rpmsave files:*

```
sudo rpmconf -a
```

If the current release is not in the Fedora repositories, download the RPM packages from https://sourceforge.net/projects/ncid/

- *Download server, gateways and client RPM Packages from SourceForge:*

```
ncid RPM Package          (server - required)
ncid-client RPM Package   (client & default output modules - optional)
ncid-gateways RPM Package (gateways - optional)
```

- *Download any optional output modules wanted from SourceForge:*

```
ncid-MODULE RPM Package  (optional client output modules)
```

- *Install or Upgrade the packages:*

  - *Using the file viewer:*

```
- Open the file viewer to view the NCID RPM packages
- Select the RPM packages
- Right click selections and select "Open with Package installer"
```

  - *Using dnf:*

```
sudo dnf install ncid\*.rpm
```

CONFIGURE:

> *The ncidd.conf file is used to configure ncidd.*

- *The default modem port in ncidd is /dev/ACM0. If you need to change it, set your modem port in ncidd.conf. This assumes serial port 0:*

```
   set ttyport = /dev/ttyS0
```

- *If you are using a Gateway to get the Caller ID instead of a local modem, you need to set cidinput:*

```
   set cidinput = 1 # Caller ID from a serial or USB device and optional
gateways
   set cidinput = 2 # Caller ID from gateways with modem support
   set cidinput = 3 # Caller ID from gateways without modem support
```

- *If you are using a local modem with or without a Gateway:*

```
   set cidinput = 0  (this is the default)
```

FIRST STARTUP:

- *If you are running the server and client on the same computer and using a modem:*

```
   sudo service ncidd start

   ncid &
```

- *If you are running the server and using a gateway:*

```
   sudo service ncidd start

   sudo service <name of gateway> start

   ncid &
```

- *Call yourself and see if it works, if not,*

> *stop the gateway used:*

```
    sudo service <name of gateway> stop
```

> *stop the server:*

```
    sudo service ncidd stop
```

> *and continue reading the test sections.*

> - *If everything is OK, enable the NCID server, gateways and client modules you are using, to autostart at boot.*

> *The GUI ncid client must be started after login, not boot.*

> *NOTE:*

>> *The ncid client normally starts in the GUI mode and there is no ncid.init script to start or stop it. There are rc.init scripts for starting ncid with output modules, for example: ncid-page, ncid-kpopup, etc.*

START/STOP/RESTART/RELOAD/STATUS:

> *Use the 'service' command to control any of the daemons. The service commands are: start, stop, restart, reload and status. The client can also be started using the output module name instead of ncid. All output modules can be run at the same time.*

> *Here are some examples:*

> - *Start the NCID server:*
>
> ```
> sudo service ncidd start
> ```

> - *Stop the sip2ncid server:*
>
> ```
> sudo service sip2ncid stop
> ```

> - *Reload the server alias file:*
>
> ```
> sudo service ncidd reload
> ```

> - *Restart ncid using ncid-page:*
>
> ```
> sudo service ncid-page start
> ```

> - *Get the status of ncid using ncid-speak:*
>
> ```
> sudo service ncid-speak status
> ```

> *Review the man page:* ***man service***

AUTOSTART:

> *Use the 'chkconfig' command to turn the service on/off for starting at boot.*

> *Here are some examples:*

> - *Autostart ncidd at boot:*
>
> ```
> sudo chkconfig ncidd on
> ```

> - *Autostart ncid-page at boot:*
>
>   ```
>   sudo chkconfig ncid-page on
>   ```

> - *Autostart ncid-kpopup at boot:*
>
>   ```
>   sudo chkconfig ncid-kpopup on
>   ```

> - *List runlevels for sip2ncid:*
>
>   ```
>   sudo chkconfig --list sip2ncid
>   ```

> - *Disable ncid-speak from starting at boot:*
>
>   ```
>   sudo chkconfig ncid-speak off
>   ```

> *Review the manpage:* ***man chkconfig***

Windows Install

> *Install either the Windows client package or the complete package.*

> *Table of Contents*

Sections:

> *WINDOWS CLIENT-ONLY INSTALL (NCID version 1.7 and newer):*
>
> > *Requirements*
> > *Install NCID Client*
> > *Upgrade*
> > *Autostart at Login*
> > *Specifying command-line arguments and options*

> *WINDOWS CLIENT-ONLY INSTALL (NCID version 1.6 and older):*
>
> > *Requirements*
> > *Install ncid.exe*

> *WINDOWS 10 and 11 COMPLETE INSTALL:*

WINDOWS CLIENT-ONLY INSTALL (NCID version 1.7 and newer):

> ***Requirements***

> - *Windows client version 1.7 and newer*

> - *NCID server*

> - *ActiveTcl-8.6.9 and newer*

- *Go to the ActiveTcl download page*

- *Download of the ActiveTcl installer requires a free account.*

- *Run the ActiveTcl installer and for **Choose Setup Type** you must select **Complete**. Accept all other defaults.*

- *Python3*

  - *Go to Python Releases for Windows and choose "Download Windows installer".*
    Install the full package and check "Use admin privileges when installing py.exe" and "Add Python.exe to PATH".
    Do not use the Windows store to obtain the Python package because tcl will not be able to find Python.

  - *After installing python, install the following using pip:*

    ```
    pip install pytz phonenumbers pydialog
    ```

Choose only ONE of the following to install:

- *Zenity for Windows version 3.20.0, dated August 11, 2016 (on the Wayback Machine)*

Even though this has an older date than WinZenity below, it is recommended because it has more features. Be sure to read this reported issue before installing; the NCID Developers did not experience this issue.

or

- *WinZenity (Zenity version 3.6 Portable for Windows), dated February 10, 2020*

There is no installer, simply unzip zenity.zip and put the extracted zenity.exe in a directory where it can be found. Suggest <drive>\ncid\zenity.exe.

Install NCID Client

- *Download "ncid-VERSION-client_windows_setup.exe" from SourceForge.*

- *Run the installer:*

  - *Decide if you want a desktop link (default).*

  - *Decide if you want a startup link (optional; can be added manually after installation).*

  - *Accept defaults for all other prompts until you get to the server address.*

  - *Change the server address if different from the default shown.*

Examples:

```
192.168.22.10
```

```
ncid.sourceforge.net
```

- *The default installation folder is:*

```
C:\ncid
```

- *File ncid.conf will be located in the same folder as ncid.tcl. Feel free to edit this file and change options as desired.*

- *To run the client from the Windows Command Prompt, simply type ncid.tcl followed by command-line arguments and options. No Windows shortcut is needed.*

- *To run ncid.tcl from Windows Explorer, make sure ncid.conf has a correct "set Host" line, then double-click on ncid.tcl.*

Upgrade

- *Install the latest ncid client for windows and it will replace the old version.*

If an existing ncid.conf is detected it will be preserved; the one for the new version will be called ncid.newconf.

- *Install the latest ActiveTcl version, must be version 8.6.9 and newer.*

Autostart at Login

These steps assume you have successfully completed the Install NCID Client steps.

You can make ncid autostart at login by simply copying the ncid shortcut from the desktop to your Startup folder. Follow these steps:

1. *Find the newly-installed ncid shortcut (icon) on the desktop.*
2. *Right-click on it and choose Copy.*
3. *Click on Start and choose All Programs.*
4. *Scroll until you find the folder called Startup.*
5. *Right-click on Startup and choose Open.*
6. *Paste the shortcut with ctrl-v, or in the menu bar choose Edit->Paste.*

Specifying command-line arguments and options

Command-line arguments and options can be added to a shortcut. The install package creates a desktop shortcut that contains the server address argument, for example:

```
C:\ncid\ncid.tcl 192.168.22.10
```

The default port number is 3333 but it can also be changed in the shortcut if it is different from the default of 3333. Right-click on the shortcut, choose Properties, then add a space followed by the port number, for example:

```
C:\ncid\ncid.tcl 192.168.22.10 3334
```

Adding options requires they be specified before the server's IP address or hostname.

> *You should add two hyphens (--) right after ncid.tcl so that the TCL/TK interpreter does not confuse options intended for itself and those intended for ncid.*

> *You can mix short and long options.*

> *Example:*

```
C:\ncid\ncid.tcl -- -D 120 --ring 5 -X -H 192.168.22.10 3334
```

> *Supported options:*

```
--alt-date,                  -A
--delay <seconds>,           -D <seconds>
--hostname-flag,             -H
--noexit,                    -X
--PopupTime <1-99 seconds>,  -t <1-99 seconds>
--ring <0-9|-1|-2|-3|-4|-9>, -r <0-9|-1|-2|-3|-4|-9>
```

> *Unsupported options:*

```
--no-gui
--pidfile,                   -p <file>
--module,                    -P <module name>
--verbose,                   -v <1-9>
--version,                   -V
--wakeup,                    -W
```

WINDOWS CLIENT-ONLY INSTALL (NCID version 1.6 and older):

> ***Requirements***

- *Windows client version 1.6 and older*
- *NCID server*

> ***Install ncid.exe (deprecated, uses FreeWrap)***

> *The following steps have been deprecated but are being kept for historical purposes. This older Windows client is still available for download from SourceForge for NCID versions 1.6 and older.*

> *The older version used freeWrap to bundle the ncid client script and the needed TCL/TK interpreter. This resulted in a simple, stand-alone executable. Unfortunately, it had a technical limitation where the ncid.conf file could not be edited. Although it was still possible to specify command line arguments via the Windows shortcut, not all ncid.conf settings had command line equivalents, which meant their default settings could not be changed. This issue has been eliminated in the current windows install.*

> *Here then are the Windows client instructions for NCID version 1.6 and older....*

- *Execute the ncid installer:*
  - *NCID Versions 1.4 and older:*

```
ncid-VERSION-client_setup.exe
```

- *NCID Versions 1.5 and newer:*

```
ncid-VERSION-client_win10_x64_setup.exe
```

Examples:

```
ncid-1.0-client_setup.exe

ncid-1.6-client_win10_x64_setup.exe
```

- *You'll be asked for the NCID server address. The default is 127.0.0.1, so you will need to change it.*

Examples:

```
192.168.22.10

ncid.sourceforge.net
```

WINDOWS 10 and 11 COMPLETE INSTALL:

The Windows Subsystem for Linux (WSL) is a full compatibility layer for running Linux applications on Windows.

- *You must be running Windows 10 version 1607 (the Anniversary update) or Windows 11 22H2.*
- *WSL only runs on 64-bit versions; 32-bit versions are not supported.*

Step-by-step screenshot guide to show you how to install bash on Windows 10 and 11.

Be warned, you are trail blazing. The ncid packages were not tested.

Obtaining Caller ID

Table of Contents

Description

Description

NCID requires hardware to obtain Caller ID. It can be a supported modem, a supported device, or a gateway.

Devices Supported
Modems
Gateways

Devices Supported

Table of Contents

Devices Index

- *Modems*

> - *ATA (Analog Terminal Adapter)*
> - *ARTECH AD102 USB device*
> - *CID Easy USB devices*
> - *CTI Comet USB device*
> - *Holtek HT9032D based PSTN Caller ID module*
> - *NetCallerID serial device*
> - *Obihai VoIP Telephone Adapters and IP Phone*
> - *Tel-Control, Inc. (TCI) serial devices*
> - *Whozz Calling serial devices*
> - *Whozz Calling Ethernet Link devices*

Modems

> *Modems are normally used to obtain the Caller ID. They can also be used dial a number or hangup on a call. NCID supports up to 5 modems or serial devices.*

> *Any Caller ID serial or USB modem supported by the operating system can be used. See Incomplete list of working modems.*

> *NCID can also use those rare modems that do not support Caller ID by configuring gencid in* ***ncidd.conf****, but such modems are limited to indicating the date and start time of the calls. If a Gateway is used to obtain the Caller ID these modems can be used to dial a number and hangup on a call.*

> *See the modem Configuration section for information on configuring modems.*

ATA (Analog Terminal Adapter)

> *The ATA hardware is for VoIP (Voice over Internet Protocol).*

> *VoIP telephone services use an Analog Terminal Adapter, sometimes called a VoIP gateway.*

> *See also Voip-info.org: A reference guide to all things VOIP.*

> *In order to receive Caller ID from VoIP, the local network must be configured. Three configurations are considered here:*

> - *One device: Cable/DSL Modem with integrated ATA device*
> - *Two devices: Cable/DSL Modem + Router Switch with integrated ATA device*
> - *Three devices: Cable/DSL Modem + Router Switch + ATA device*

> ***One device: Cable/DSL Modem with integrated ATA device***

> *Many cable companies such as Comcast and Time Warner now offer bundled services, referred to in the industry as "triple play service." This delivers television, Internet service and digital phone service via a single device.*

> *The protocol used for the digital phone service is usually proprietary.*

> *NCID is not supported in this configuration.*

> ***Two devices: Cable/DSL Modem + Router Switch with integrated ATA device***

These router and ATA combo devices may be configured to put the Caller ID on the built-in switch. If you have other routers working, please contribute to this list:

Router	***Model***	***Settings***	***Configuration***
Linksys	*WRTP54G*	*-*	*(has "P" in model name)* *use Vonage Talk*
Linksys	*RT31P2*	*DMZ*	*put computer IP address in the DMZ*

Three devices: Cable/DSL Modem + Router Switch + ATA device

A stand-alone ATA device connected to your network will make its Caller ID info (Session Initiation Protocol, or SIP) available to all the other network devices that are listening for it. A typical setup has the ATA connected to one physical port on the router/switch and the computer running NCID is connected to a different physical port. Most modern routers/switches isolate the network traffic on any one physical port from all the other physical ports. This is done on purpose to optimize network traffic throughput and provide better performance.

The problem is that having the network traffic isolated in this way does not allow the NCID computer to ever receive the Caller ID info from the ATA.

To circumvent this problem, you have several options:

A. Use an Ethernet Tap.

This is the preferred method to obtain Caller ID. The USB Powered 5-Port 10/100 Ethernet Switch TAP by Dualcomm is a good choice and has been successfully used with NCID. The Dualcomm USB powered 5-port Ethernet Switch TAP mirrors all ethernet traffic on port 1 to port 5. Simply plug your ATA into port 1 and your NCID server into port 5.

- *The NCID server and ATA need to be (relatively) close together in order to connect directly to the ethernet TAP.*
- *Requires no software configuration beyond the sip2ncid setup.*
- *Requires additional hardware.*

B. Use port mirroring.

Port mirroring is not port forwarding.

- *Requires DD-WRT, OpenWRT, or similar Linux-based OS to be running on your home router.*
- *Requires manual configuration of the port mirror on your home router.*
- *Any modification to the firewall rules or QoS settings in DD-WRT will result in the port mirroring commands being discarded; you will either have to reboot DD-WRT or manually enter the commands via SSH to restart the port mirror.*
- *The NCID server and ATA can be located anywhere on your home network.*
- *No additional hardware needed.*

STEPS TO CONFIGURE DD-WRT

1. Use ssh to connect to your router and enter the following port mirroring commands, substituting your values for ip-of-sip-ata and ip-of-ncid-server:

```
iptables -t mangle \
         -A POSTROUTING \
```

```
         -d ip-of-sip-ata \
         -j ROUTE \
         --tee --gw ip-of-ncid-server

iptables -t mangle \
         -A PREROUTING \
         -s ip-of-sip-ata \
         -j ROUTE \
         --tee --gw ip-of-ncid-server
```

2. To verify the port mirror is setup properly, use:

```
iptables -t mangle -L -v -n
```

which will provide output that should show something similar to:

```
Chain PREROUTING (policy ACCEPT 4510K packets, 2555M bytes)
pkts bytes target prot opt in out source destination
....
219 152K ROUTE 0 -- * * ip-of-sip-ata 0.0.0.0/0 ROUTE gw:ip-of-ncid-
server tee
....

Chain POSTROUTING (policy ACCEPT 17M packets, 7764M bytes)
pkts bytes target prot opt in out source destination
....
206 82184 ROUTE 0 -- * * 0.0.0.0/0 ip-of-sip-ata ROUTE gw:ip-of-ncid-
server tee
....
```

3. Follow the sip2ncid setup instructions to make sure that SIP packets are being received.

4. When everything is working properly, add the port mirroring commands to the DD-WRT startup commands in the Management tab so that they will be run whenever DD-WRT is rebooted.

C. Use Ettercap.

Convince your router to send all SIP packets to your NCID server and have your NCID server pass the packets on to your ATA. This is most easily and robustly accomplished through the use of ettercap.

- *If the NCID server or ettercap fails, your router and SIP ATA will automatically start communicating directly within a few minutes as the SIP ATA and router are not physically isolated.*
- *The NCID server and ATA can be located anywhere on your home network.*
- *No manual configuration of router is required.*
- *No additional hardware needed.*

STEPS TO CONFIGURE ETTERCAP

Perform these steps from a command prompt on your NCID server.

1. To determine the proper interface for ettercap to use, `ifconfig` will show all available interfaces. For example, wired ethernet is eth0 and wireless ethernet is wlan0 on raspios.

2. Install ettercap.

- *if on Ubuntu, raspios and other Debian-based systems:*

```
sudo apt-get install ettercap-text-only
```

- *if on Fedora and other Redhat-based systems:*

```
sudo dnf install ettercap
```

3. Execute ettercap, substituting your values for interface, ip-of-sip-ata and ip-of-home-router. The forward slashes are mandatory.

- *if using IPV4, surround each IP address with one leading slash and one trailing slash:*

```
sudo ettercap -T -D -i interface -M arp:remote \
              /ip-of-sip-ata/ /ip-of-home-router/
```

- *if using IPV6, surround each IP address with two leading slashes and one trailing slash:*

```
sudo ettercap -T -D -i interface -M arp:remote \
              //ip-of-sip-ata/ //ip-of-home-router/
```

4. Follow the sip2ncid setup instructions to make sure that SIP packets are being received.

5. You will want to add ettercap to your operating system startup sequence. Steps to do this vary depending on distribution and even depending on the version of a specific distribution. Consult your operating system documentation on how to do this.

D. Install SIP client on NCID server.

If none of the above options are possible on your network, a SIP client can be installed on the NCID server to attract incoming call information to the NCID server. It is best practice to create a new extension number for the NCID server's SIP client and for access control to be configured on the voice gateway to prevent this extension from dialing out. Multiple command-line SIP clients are available, but it should be simple to install and use Linphone.

STEPS TO CONFIGURE LINPHONE

Perform these steps from a command prompt on your NCID server.

1. Install linphone.

- *if on Ubuntu, raspios and other Debian-based systems:*

```
sudo apt-get install linphone
```

- *if on Fedora and other Redhat-based systems:*

```
sudo dnf install linphone
```

2. Execute linphone.

```
linphonecsh init
```

```
linphonecsh register \
            --host <ip of gateway> \
            --username <extension number/username> \
            --password <password>
```

3. Follow the sip2ncid setup instructions to make sure that SIP packets are being received.

E. Use an Ethernet hub.

(Historical, not recommended)

Ethernet hubs pre-date Ethernet switches and do not isolate network traffic between physical ports. Ethernet switches have largely rendered Ethernet hubs obsolete. Some Ethernet hubs manufactured today are actually Ethernet switches in disguise. See the hub reference to determine if a hub is really a hub.

F. Use a router that supports SIP ALG (Application-level gateway).

(Historical, not recommended)

Unfortunately, not all routers implement ALG correctly. The following routers are known to use ALG properly with NCID. If you have other routers working, please contribute to this list:

Router	Model	Settings	Configuration
Linksys	WRT54G	-	(no "P" in model name) SIP packets on port 5060 may need a firmware update if the firmware version is below 1.00.6. See this link for firmware info.
Linksys	RVS4000	L2 Switch	Assuming gateway is port #1 and NCID SIP gateway is monitoring port #2: mirror port #1 to port #2

ARTECH AD102 USB device

The ARTECH AD102 is a Caller ID device with USB connectivity. This product is powered entirely from the USB port. It is a hardware-only product and can be used with your NCID software to track incoming Caller ID (including Call Waiting Caller ID) and outgoing dialing detection.

This device is sensitive to the signal levels and timings present on a country's phone lines and will likely need adjusting via artech2ncid.conf. Feel free to submit a Support Request and we'll do our best to help.

When connected to a computer, the device registers as a Human Interface Device (HID). No Linux software is provided by ARTECH to interface with the AD102 but fortunately an NCID developer was able to reverse engineer the complex protocol.

The ncidd server must be configured to use it. The server normally assumes a modem is going to be used so it must be configured to use a gateway instead.

Refer to the artech2ncid setup in the Gateways section to configure NCID to work with the ARTECH AD102.

CID Easy USB devices

> *The CID Easy Model-E supports up to two different analog lines and the CID Easy Model-F supports up to four. They detect incoming Caller ID only. Both are powered entirely from the USB port and both are a hardware-only product.*

> *When connected to a computer, the device registers as a Human Interface Device (HID). The device manufacturer has documented the communications protocol used.*

> *The ncidd server must be configured to use it. The server normally assumes a modem is going to be used so it must be configured to use a gateway instead.*

> *Refer to the cideasy2ncid setup in the Gateways section to configure NCID to work with the CID Easy devices.*

CTI Comet USB device

> *The CTI Comet USB is a Caller ID device with USB connectivity. This product is powered entirely from the USB port. It is a hardware-only product and can be used with your NCID software to track incoming calls.*

> *A traditional modem communicates with ncidd using ASCII text data, but the CTI Comet USB uses binary data so ncidd cannot monitor it directly. Instead, the CTI Comet USB is monitored by the **xdmf2ncid** gateway.*

> *The ncidd server must be configured to use it. The server normally assumes a modem is going to be used so it must be configured to use a gateway instead.*

> *Refer to the xdmf2ncid setup in the Gateways section to configure NCID to work with the CTI Comet USB.*

Holtek HT9032D based PSTN Caller ID module

> *The Holtek HT9032D based PSTN Caller ID module is a Caller ID device with USB connectivity achieved by the USB to UART TTL cable adapter. As such this product is powered entirely from the USB port. It can be used with your NCID software to track incoming calls.*

> *A traditional modem communicates with ncidd using ASCII text data, but the Holtek HT9032D based PSTN Caller ID module uses binary data so ncidd cannot monitor it directly. Instead, the Holtek HT9032D based PSTN Caller ID module is monitored by the **xdmf2ncid** gateway.*

> *The ncidd server must be configured to use it. The server normally assumes a modem is going to be used so it must be configured to use a gateway instead.*

> *Refer to the xdmf2ncid setup in the Gateways section to configure NCID to work with the Holtek HT9032D based PSTN Caller ID module.*

NetCallerID serial device

> *The NetCallerID device is used in place of a modem. It is no longer manufactured by Ugotcall (archived info here and here) but you can sometimes find it on eBay.*

> *The ncidd server must be configured to use it. The server normally assumes a modem is going to be used so it must be configured to use a serial NetCallerID device that does not use AT commands.*

> *Uncomment these lines in **ncidd.conf** (this assumes the device is connected to serial port 0):*

```
# set ttyport = /dev/ttyS0    # Linux Serial Port 0
# set ttyspeed = 4800         # NetCallerID port speed
```

```
# set cidinput = 1
```

Here are the specifications of the NetCallerID device:

ttyport:

```
4800 8N1
```

Output Format:

```
###DATE08082225...NMBR14075551212...NAMEJOHN+++\r
###DATE...NMBR...NAME    -MSG OFF-+++\r
```

Obihai VoIP Telephone Adapters and IP Phone

Obihai sells the OBi1032 IP phone and the popular OBi100, OBi110, OBi200, OBi202 VoIP telephone adapters.

OBi devices equipped with a USB port also support the OBiLINE FXO to USB Phone Line Adapter. The OBiLINE provides PSTN (or POTS) connectivity to phones attached to the OBi device as well as to calls bridged from VoIP services to a land-line service via the OBi.

It appears that OBi devices require at least one third party VoIP service provider, even if you intend to use a POTS line as your primary incoming and outgoing service.

The OBi110 comes with an FXO port built-in.

Only the OBi100, OBi110, OBi200 with OBiLINE and OBi202 with OBiLINE, were available for development and testing. The other OBi products may work completely, partially, or not at all.

Refer to the obi2ncid setup in the Gateways section to configure NCID to work with the Obihai device.

Tel-Control, Inc. (TCI) serial devices

TCI caller ID units are used in place of one or more modems. The company is no longer in business, but you can sometimes find units on eBay or at telephone equipment liquidators. As an alternative, Whozz Calling serial devices are direct replacements for TC-1041, MLX-41, TC-1082 and MLX-42 units. NCID has been tested with a TC-1041.

Configure the TCI unit to use a baud rate of 9600. There are two banks of switches located on the front of the unit labeled S1 and S2. On bank S2 you want to set DIP switch 1 and 2 to both be ON for a 9600 baud rate.

The ncidd server must be configured to use it. The server normally assumes a modem is going to be used so it must be configured to use a serial device that does not use AT commands.

*Uncomment these lines in **ncidd.conf** (this assumes the device is connected to serial port 0):*

```
# set ttyport = /dev/ttyS0    # Linux Serial Port 0
# set ttyspeed = 9600         # TCI serial device port speed
# set cidinput = 1
```

*TCI units supply their own line id to ncidd as a two-digit number. When a setting for lineid in **ncidd.conf** is not given, ncidd will automatically replace the default with this two-digit number.*

> *You may, if you wish, prefix the two-digit number with a meaningful identifier, such as 'MLX-', by uncommenting this line in* ***ncidd.conf***

```
# set lineid = POTS
```

> *and changing it to*

```
set lineid = MLX-
```

> *Here are the specifications of the TCI serial device:*

> *ttyport:*

```
9600 8N1
```

> *Output Format is fixed field with total length of 70 bytes:*

```
01      9/05     11:17 AM       702-555-1145           CARD SERVICES
02      9/05      2:00 PM            PRIVATE
```

Whozz Calling serial devices

> *Whozz Calling serial devices are used in place of one or more modems. They are currently supported by NCID only when the Output Format switch is set to TCI.*

> *Set DIP switch 5 to OFF for a baud rate of 9600.*

> *Set DIP switch 7 to ON for TCI output format.*

> *The ncidd server must be configured to use it. The server normally assumes a modem is going to be used so it must be configured to use a serial device that does not use AT commands.*

> *Uncomment these lines in* ***ncidd.conf*** *(this assumes the device is connected to serial port 0):*

```
# set ttyport = /dev/ttyS0    # Linux Serial Port 0
# set ttyspeed = 9600         # TCI serial device port speed
# set cidinput = 1
```

> *The TCI output format supplies its own line id to ncidd as a two-digit number. When a setting for lineid in* ***ncidd.conf*** *is not given, ncidd will automatically replace the default with this two-digit number.*

> *You may, if you wish, prefix the two-digit number with a meaningful identifier, such as 'WC-', by uncommenting this line in* ***ncidd.conf***

```
# set lineid = POTS
```

> *and changing it to*

```
set lineid = WC-
```

> *Here are the specifications of the Whozz Calling serial device in TCI mode:*

> *ttyport:*

```
9600 8N1
```

Output Format is fixed field with total length of 70 bytes:

```
01      9/05     11:17 AM       702-555-1145           CARD SERVICES
02      9/05      2:00 PM            PRIVATE
```

Whozz Calling Ethernet Link

A Whozz Calling (WC) Caller ID and Call monitoring unit is used in place of one or more modems. There are various models that all monitor incoming calls and some can monitor outbound as well.

See CallerID.com.

Refer to the wc2ncid setup in the Gateways section to configure NCID to work with the WC device.

Modems

Table of Contents

Index

NCID Modem Requirements

The modem must be supported by the Operating System (Linux, FreeBSD, Macintosh, etc.) where NCID is running.

The absolute minimum modem feature that NCID requires is a modem that indicates RING, even if it does not support Caller ID. By default, if ncidd does not detect Caller ID by ring number two, it will generate a pseudo-Caller ID by setting the number to "RING" and the name to "No Caller ID". This default behavior can be disabled.

The ideal minimum is a modem that supports Caller ID.

A modem supporting FAX and/or VOICE modes is required for the optional FAX and ANNOUNCE hangup features respectively.

Some modems come configured for the US. If you live in a different country and your modem does not work, check the ncidd.log file for the country code. It will give either the bare code or the code and country, for example:

> *Modem country code: B5 United States*

> *See the Modem Country Codes section for a list of country codes.*

> *You can use **minicom** to set your country code. It only needs to be set once. Do not include it in ncidd.conf.*

> *For example, if you live in the UK, you would issue this command using **minicom**:*

> > *AT+GCI=B4*

Configuration

> *The tty port of the first modem is set in **ncidd.conf**. NCID packages for specific distributions (Fedora, Ubuntu, etc.) are usually pre-configured for an appropriate default port. You can change the default by editing **ncidd.conf** and simply uncommenting the appropriate ttyport line, or add a new port.*

> *NCID supports up to 5 modems or serial devices. See the Additional Modems section.*

```
The default tty ports:
    FreeBSD:            /dev/cuaU0
    Linux USB modem 0:  /dev/ttyACM0
    Mac OS X USB modem: /dev/cu.usbmodem24680241
```

```
The tty ports in **ncidd.conf**
    # Mac OS X internal modem:
    set ttyport = /dev/cu.modem
```

```
    # Mac OS X USB modem (Dualcomm, Zoom):
    set ttyport = /dev/cu.usbmodem24680241
```

```
    # Serial Port 0:
    set ttyport = /dev/ttyS0
```

```
    # Linux USB modem 0:
    set ttyport = /dev/ttyACM0
```

Additional Modems

> *The first modem or serial device is configured by setting options in the **ncidd.conf** file. Additional modems and serial devices can be configured in additional config files, one per modem, with their file names listed in **ncidd.conf**. As many as 5 modems and/or serial devices can be configured.*

> > *set addedmodems = "modem2.conf modem3.conf modem4.conf modem5.conf"*

> *NOTE: This list of file names must be in quotes.*

> *Each modem config file will define options for one additional modem or serial device and should specifiy at least a unique ttyport and lineid.*

> > *set lineid = line2*
> > *set ttyport = /dev/ttyACM2*

> *The lineid will be given to clients for incoming calls, and it can be specified by the client to choose an outgoing line for the DIAL feature.*

Distinctive Ring (DR)

> *From: DISTINCTIVE RING & MODEMS*

> *You can find out whether your modem supports DR by connecting to its COM port via Windows HyperTerminal (or using the Unix program **minicom**) and issuing the appropriate AT command; if it responds with an "OK", it does, otherwise it does not.*

> *The AT command to enable DR depends on the modem chipset:*

Chipset	Command
3Com/USR/TI	ATS41=1
Rockwell/Conexant	AT-SDR=7
Lucent/Agere	AT+VDR=1,0

> *Each chipset reports DR differently:*

> > *3Com reports ring codes:*

> > > ```
> > > RING A, RING B, RING C
> > > ```

> > *Rockwell reports ring codes:*

> > > ```
> > > RING 1, RING 2, RING 3
> > > ```

> > *Lucent reports the actual ring cadence (the duration of the ringing and the silent periods) with DROF/DRON messages.*

> *As an example, one long ring usually indicates the main phone number and two short rings usually indicates the first DR phone number. Responses below were generated by two short rings:*

> *3Com:*

> > *RING B*

> *Rockwell:*

> > *RING 2*

> *Lucent:*

> > *DRON=5*
> > *DROF=11*
> > *DRON=5*
> > *DROF=34*

Modem Caller ID Test

> *Start ncidd in debug mode with verbose level 3. You do not need to start any client.*

> > ***ncidd -Dv3***

> *You should get something similar to the following. The important part is the last line: `Modem is fd x` where `x` is usually a number less than 10*

> ```
> Started: 05/04/2011 21:48:14
> Server: ncidd (NCID) 0.81.15
> ```

```
logfile: /var/log/ncidd.log
Command line: ncidd
              -Dv3
Processed config file: /etc/ncid/ncidd.conf
Verbose level: 3
Configured to send 'cidlog' to clients.
Configured to send 'cidinfo' to clients.
Processed alias file: /etc/ncid/ncidd.alias
Leading 1 from a call required in an alias definition
CID logfile: /var/log/cidcall.log
CID logfile maximum size: 1000000 bytes
Data logfile: /var/log/ciddata.log
Telephone Line Identifier: -
TTY port opened: /dev/ttyACM0
TTY port speed: 19200
TTY lock file: /var/lock/lockdev/LCK..ttyACM0
TTY port control signals enabled
Caller ID from a modem and optional gateways
Handles modem calls without Caller ID
Sent Modem 20 of 20 characters:
AT Z S0=0 E1 V1 Q0
Modem response: 26 characters in 1 read:
AT Z S0=0 E1 V1 Q0
OK
Try 1 to init modem: return = 0.
Modem initialized.
Sent Modem 11 of 11 characters:
AT+VCID=1
Modem response: 17 characters in 1 read:
AT+VCID=1
OK
Modem set for CallerID.
Network Port: 3333
Debug Mode
Not using PID file, there was no '-P' option.
Modem is fd 4
```

If you did start a client, you will get a line something like:

```
Client 6 from 127.0.0.1 connected, sent call log: /var/log/cidcall.log
```

Next, call yourself and you should see something like:

```
RING
CIDINFO: *LINE*VOIP*RING*1*TIME*21:49:49*
DATE = 0504
TIME = 2149
NMBR = 4075551212
NAME = Chmielewski Joh
CID: *DATE*05042011*TIME*2149*LINE*VOIP*NMBR*4075551212*MESG*NONE*NAME*John*
RING
CIDINFO: *LINE*VOIP*RING*2*TIME*21:49:55*
```

Hang up the phone and a second or two later you should see:

```
CIDINFO: *LINE*VOIP*RING*0*TIME*21:50:02*
```

Do a <CTRL-C> to break out and end the test:

```
^CReceived Signal 2: Interrupt
Terminated: 05/04/2011 21:53:30
```

If your modem does not support CID, ncidd will generate a CID line on ring 2 to indicate "No Caller ID" by setting the number to "RING" and the name to "No Caller ID". You will get something like:

```
RING
CIDINFO: *LINE*VOIP*RING*1*TIME*21:53:02*
RING
CIDINFO: *LINE*VOIP*RING*2*TIME*21:53:08*
CID: *DATE*05042011*TIME*2153*LINE*VOIP*NMBR*RING*MESG*NONE*NAME*No Caller ID*
RING
CIDINFO: *LINE*VOIP*RING*3*TIME*21:53:14*
```

*If ncidd is configured to not generate a CID line for "No Caller ID" (**ncidd.conf** has gencid=0), you will get something like:*

```
RING
CIDINFO: *LINE*VOIP*RING*1*TIME*21:53:02*
RING
CIDINFO: *LINE*VOIP*RING*2*TIME*21:53:08*
RING
CIDINFO: *LINE*VOIP*RING*3*TIME*21:53:14*
```

Hang up the phone and a second or two later you should see:

```
CIDINFO: *LINE*VOIP*RING*0*TIME*21:53:21*
```

Do a <CTRL-C> to break out and end the test:

```
^CReceived Signal 2: Interrupt
Terminated: 05/04/2011 21:53:30
```

Modem Commands that configure Caller ID

AT#CID=1

> *Enables Caller ID in USR, Texas Instruments, Rockwell compatible modems (excluding software modems and Rockwell HCF), Hayes, several Pace modems, PowerBit, GVC, PCTel, IDC (VR series) devices, Diamond Supra (Rockwell compatible).*

AT+VCID=1
or
AT+FCLASS=8;+VCID=1

> *Enables Caller ID in all IS-101 modems, Lucent LT, Rockwell HCF (V.90 or K56FLEX, e.g. PCI modems from Creative), some Pace modems (IS-101 compatible), Multitude, IDC, Cirrus Logic, most IDC modems.*

> ***AT#CLS=8#CID=1***

> *Enables Caller ID in voice mode on some 56K USR modems, some Rockwell compatible (Boca Research, Cardinal, voice Zoom).*

> ***AT#CC1***

> *Enables Caller ID on older non-voice Aspen modems, older Cirrus Logic, Motorola Voice Surfer, Phoebe.*

> ***AT*ID1***

> *Enables Caller ID on some Motorola modems.*

> ***AT%CCID=1***
> *or*
> ***AT%CCID=3***

> *Enables Caller ID on Practical Peripherals modems.*

> ***AT$JSCID=4,1***
> *or*
> ***AT$JSCD=1,0***

> *Enables Caller ID on ELSA ML 56k Internet II (Netherlands) modems.*

> ***AT#CID=1***
> *or*
> ***AT+VCID=1***

> *Enables Caller ID on most modems not listed above.*

> ***AT+FCLASS=8***
> *or*
> ***AT+FCLASS=8;+VCID=1***
> *or*
> ***AT-STE=1;+VCID=1***

> *Generic commands to select Active Service Class (Voice Mode) on most voice modems.*

> ***AT+VRID=0***

> *The other AT commands above enable Caller ID during an incoming call. This command reports the most recently received Caller ID after a call has completed. It can sometimes be useful in testing.*

Modem Country Codes

> ***U.S. Robotics Country Codes***

> *This extended syntax command selects and indicates the country of operation for the modem. It determines the settings for any operational parameters that need to be adjusted for national regulations or telephone networks.*

> ***Syntax***

> ***+GCI=CountryCode***

> ***Defined values for supported countries are:***

Country	Code	Country	Code	Country	Code
Austria	0A	Iceland	52	Poland	8A
Belgium	0F	Ireland	57	Portugal	8B
Cyprus	2D	Israel	58	Spain	A0
Czech Republic	2E	Italy	59	Slovakia	FB
Denmark	31	Latvia	F8	Slovenia	FC
Estonia	F9	Liechtenstein	68	Sweden	A5
Finland	3C	Lithuania	F7	Switzerland	A6
France	3D	Luxembourg	69	TBR-21(Default)	F6
Germany	42	Malta	B4	Turkey	AE
Greece	46	Netherlands	7B	United Kingdom	B4
Hungary	51	Norway	82	United States	B5

> *The factory default is F6 indicating TBR-21.*

> *TBR-21 is a European telecommunications standard to which all telephone equipment must adhere to, to be allowed connection to Europe's public switched telephone network. It is the default even for U.S. Robotics modems sold in the United States.*

> ***Command to report current country code value***

> ***+GCI?***

> ***COMMAND RESPONSE:***

```
+GCI: CountryCode
```

> ***RESPONSE EXAMPLE FOR FRANCE:***

```
+GCI: 3D
```

> ***Command to report all supported country code values***

> ***+GCI=?***

> ***COMMAND RESPONSE:***

```
+GCI: (CountryCode[,CountryCode][,CountryCode])
```

> ***RESPONSE EXAMPLE:***

> *This indicates the modem has been set for use in the United Kingdom, France or Germany.*

```
+GCI: (B4,3D,42)
```

> ***ACM5003-M Modem Country Code List (subset of the T.35 Country Code List used by most modems)***

Code	Country	Code	Country	Code	Country
00	Japan	53	India	9C	Singapore
07	Argentina	54	Indonesia	9F	South Africa
09	Australia	57	Ireland	A0	Spain
0A	Austria	58	Israel	A1	Sri Lanka
0F	Belgium	59	Italy	A5	Sweden
16	Brazil	61	Korea (Republic of)	A6	Switzerland
1B	Bulgaria	62	Kuwait	A9	Thailand

```
20   Canada          64   Lebanon           AD   Tunisia
25   Chile           69   Luxembourg        AE   Turkey
26   China           6C   Malaysia          B3   United Arab Emirates
27   Columbia        73   Mexico            B4   United Kingdom
2D   Cyprus          77   Morocco           B5   Unites States
2E   Czech Republic  7B   Netherlands       B7   Uruguay
31   Denmark         7E   New Zealand       B8   Russia
36   Egypt           82   Norway            F9   Estonia
3C   Finland         84   Pakistan          FA   US Virgin Islands
3D   France          89   Philippines       FB   Slovakia
42   Germany         8A   Poland            FC   Slovenia
46   Greece          8B   Portugal          FD   (Universal)
50   Hong Kong       8E   Romania           FE   Taiwan
51   Hungary         98   Saudi Arabia
52   Iceland         99   Senegal
```

Selecting Modem Country Codes

> *Not every country has a separate country code. Some countries share country codes. Here is a list of AT codes for some modems and a table of country codes to select for country locations.*

NCID Gateways

> *Table of Contents*

Gateways Index

artech2ncid setup

> *How to setup one ARTECH AD102 USB device for Caller ID using artech2ncid.*

> ***Sections:***

> ***REQUIREMENTS:***

> *The ARTECH AD102 USB device connects to a POTS (Plain Old Telephone System) line to provide Caller ID on incoming or outgoing calls. It can only handle one telephone line.*

> ***CONFIGURATION:***

> *Connect the ARTECH AD102 USB device to the computer using a USB cable and then connect the ARTECH AD102 USB device to the telephone line.*

> *The ncidd server defaults to using a modem and optional gateways to get Caller ID. If you are using a modem for Caller ID on a POTS (standard telephone) line, you can use the artech2ncid gateway to handle an additional POTS line. No need to configure* ***ncidd.conf****.*

> *If you are not using a modem or serial device for Caller ID, you need to configure ncidd by changing this line in* ***ncidd.conf****:*

```
From:       # set cidinput = 3
To:         set cidinput = 3
```

> *If you are using a modem for hangup or to dial calls, you must have a modem connected to the same telephone line as the device connected to the AD102, and you must configure ncidd by changing these lines in* ***ncidd.conf****:*

```
From:       # set cidinput = 2
To:         set cidinput = 2
```

```
From:       # set lineid = POTS
To          set lineid = <device name>
```

> *Notes:*

- *the* ***artech2ncid lineid*** *is the device name*
- *see Note 1 for an explanation of cidinput*

> *Once you change* ***ncidd.conf****, you must start/restart ncidd to read it.*

> *Normally you do not have to edit* ***artech2ncid.conf****.*

> *You may need to change other settings in* ***artech2ncid.conf****. For instance, the configuration assumes ncidd is running on the same computer using its default port.*

> ***EXAMPLE:***

> *This ncidd.conf example is for the ARTECH AD102 USB device:*

- *connected to /dev/CometUSB0 or /dev/HoltekUSB0*
- *a modem connected to the same phone line connected to the Artech AD102 USB device*
- *automatic plain hangup on unwanted calls*

```
# ARTECH AD102 USB device:
set cidinput = 2
set lineid = "ARTECH"
```

> ***TESTING:***

> *Once you connected the ARTECH AD102 USB device and modified* ***artech2ncid.conf****, if necessary, start artech2ncid:*

```
artech2ncid [--test]
```

The `--test` parameter is optional, but it is a good idea to use it so that artech2ncid does not connect to the NCID server during the configuration process. You'll want to use this parameter if this is the first time you are setting artech2ncid up:

```
artech2ncid --test
```

The above command puts artech2ncid in test and debug modes at verbose level 3. It will display verbose statements on the terminal, ending with "Waiting for calls".

If artech2ncid terminates you should be able to see why and fix it.

You can get a detailed usage message by executing:

```
artech2ncid --help
```

or print out the manual page by executing:

```
man artech2ncid
```

Call yourself. You should see more verbose messages as the call is processed. If it looks OK, terminate artech2ncid with ***<CTRL><C>****.*

Next, restart artech2ncid in debug mode so it will connect to ncidd:

```
artech2ncid -D
```

The above command puts artech2ncid in debug mode at verbose level 3. It will display verbose statements on the terminal, ending with "Waiting for calls".

Call yourself. If you do not get a Caller ID message sent to ncidd, you should get an error message saying what is wrong.

If you had Caller ID sent to a client, setup is complete.

START/STOP/RESTART/STATUS/AUTOSTART:

Normally artech2ncid is started using the provided init, service, rc, or plist script for your OS. For more information, refer to the INSTALL section for your OS. If none is provided you need to start artech2ncid manually:

```
sudo artech2ncid
```

You can also set it up to start at boot, along with ncidd. If any options are needed, add them to ***artech2ncid.conf****.*

If artech2ncid does not work, you should have enough information to ask for help.

cideasy2ncid setup

How to setup one CID Easy Model-E or Model-F USB device. Only one CID Easy device can be connected at a time.

Sections:

REQUIREMENTS:

Connect the CID Easy USB Model-E (2 ports labeled A B) or the CID Easy USB Model-F (4 ports labeled A B C D) to one or more POTS (Plain Old Telephone System) lines to provide Caller ID on incoming calls.

CONFIGURATION:

Connect the CID Easy Model-E or Model-F USB device to the computer using a USB cable and then connect the CID Easy Model-E or Model-F USB device to the telephone line using the remaining cable.

The ncidd server defaults to using a modem and optional gateways to get Caller ID. If you are using a modem for Caller ID on a POTS (standard telephone) line, you can use the cideasy2ncid gateway to handle an additional POTS or line. No need to configure ***ncidd.conf****.*

If you are not using a modem or serial device for Caller ID, you need to configure ncidd by changing this line in ***ncidd.conf****:*

```
From:       # set cidinput = 3
To:         set cidinput = 3
```

If you are using a modem for hangup or to dial calls, you must have a modem connected to the same telephone line as the CID Easy device, and you must configure ncidd by changing these lines in ***ncidd.conf****:*

```
From:       # set cidinput = 2
To:         set cidinput = 2
```

```
From:       # set lineid = POTS
To          set lineid = CIDEASY
```

Notes:

- *When the gateway is running, the* ***generated*** *lineid it transmits to ncidd uses the convention* ***<gatewayid>-<port letter>*** *where:*
  - *The* ***<gatewayid>*** *comes from* ***ncidd.conf*** *above.*
  - *The* ***<port letter>*** *is supplied automatically by the CID Easy device depending on the device model and which port detected the Caller ID.*
    The Model-E port letters would be A or B.
    The Model-F port letters would be A or B or C or D.
- *see Note 1 for an explanation of cidinput*

Once you change ***ncidd.conf****, you must start/restart ncidd to read it.*

> *Normally you do not have to edit **cideasy2ncid.conf**.*

> *You may need to change settings in **cideasy2ncid.conf**. For instance, the configuration assumes ncidd is running on the same computer using its default port.*

> ***EXAMPLE:***

> *This ncidd.conf example is for the CID Easy Model E or Model F USB device:*

- *a CID Easy USB device connected to USB and telephone line connected to port A*
- *a modem connected to the same phone line connected to the CID Easy USB device*
- *automatic plain hangup on unwanted calls*

```
# ncidd.conf
set cidinput = 2
set lineid = "CIDEASY"
```

> ***TESTING:***

> *Once you connect the CID Easy device and configure **ncid.conf**, start cideasy2ncid:*

```
sudo cideasy2ncid [--test]
```

> *The `--test` parameter is optional, but it is a good idea to use it so that cideasy2ncid does not connect to the NCID server during the configuration process. You'll want to use this parameter if this is the first time you are setting cideasy2ncid up:*

```
sudo cideasy2ncid --test
```

> *The above command puts cideasy2ncid in test and debug modes at verbose level 3. It will display verbose statements on the terminal, ending with "Waiting for calls". It should show the USB port for the device.*

> *If cideasy2ncid terminates you should be able to see why and fix it.*

> *You can get a detailed usage message by executing:*

```
cideasy2ncid --help
```

> *or print out the manual page by executing:*

```
man cideasy2ncid
```

> *Call yourself. You should see more verbose messages as the call is processed. If it looks OK, terminate cideasy2ncid with **<CTRL><C>**.*

> *Next, restart cideasy2ncid in debug mode so it will connect to ncidd:*

```
sudo cideasy2ncid -Dv3
```

> *The above command puts cideasy2ncid in debug mode at verbose level 3. Call yourself. If you do not get a Caller ID message sent to ncidd, you should get an error message saying what is wrong.*

> *If you had Caller ID sent to a client, setup is complete.*

> ***START/STOP/RESTART/STATUS/AUTOSTART:***

> *Normally cideasy2ncid is started using the provided init, service, rc, or plist script for your OS. For more information, refer to the INSTALL section for your OS. If none is provided you need to start cideasy2ncid manually:*

```
    sudo cideasy2ncid
```

> *You can also set it up to start at boot, along with ncidd. If any options are needed, add them to* ***cideasy2ncid.conf****.*

> *If cideasy2ncid does not work, you should have enough information to ask for help.*

email2ncid setup

> *How to setup the email-to-NCID message gateway to convert an email into an NCID message and send it to the NCID server. If the notify option is used, it will only send the email subject line to the NCID server.*

> ***Sections:***

> *REQUIREMENTS*
> *CONFIGURATION*
> *TESTING*
> *STEP-BY-STEP SETUP FOR RASPBERRY Pi*

> ***REQUIREMENTS:***

- *A dynamic DNS service. Here are just a few examples:*

Name	*Basic Service*	*Website*
ChangeIP	*free*	*https://www.changeip.com/dns.php*
DNSdymanic	*free*	*https://www.dnsdynamic.org/*
Dynu	*free*	*https://www.dynu.com*
DynDNS	*paid*	*https://www.dyn.com*

- *A Mail Transport Agent (MTA):*

> *exim, postfix, sendmail, etc*

- *procmail*

> ***CONFIGURATION:***

- *firewall:*

> > *Forward port 25 TCP/UDP to the computer running the MTA.*

- *procmail:*

> *Run the following setup script to create or update $HOME/.procmailrc:*

```
    ncid-setup email2ncid
```

> *The $HOME/.procmailrc file is configured to pipe to **email2ncid** when the subject line is **NCID Message**. A commented out recipe will pipe to **email2ncid --notify** if the From: email address matches. A second commented out recipe will forward the email if the From: email address matches.*

- *MTA:*

  - *Accept mail for your dynamic DNS service host name.*
  - *Listen on all network interfaces, not just localhost.*
  - *Use mbox format.*
  - *Configure smarthost to your server provider email host if you want to send email.*

TESTING:

You can test if email2ncid is configured for the email server and can connect to it. The test line and result should be similar to:

```
   $ echo test | email2ncid -t3
```

test test=3 Configuration File: /etc/ncid/email2ncid.conf mesg=0 start=0 plain=0 html=0 multi=0 meta=0 status=0 ncidserver: localhost:3334 200 Server: ncidd (NCID) 1.5 210 API: 1.3 Feature Set 1 2 3 4 253 No Call log

*Next, send an email to yourself at the host name you picked for the Dynamic DNS service. The subject line must be: **NCID Message***

If it does not work, save the email message. You can retest it by:

cat saved_email_message | email2ncid -t1

Review email2ncid.1 for more information.

STEP-BY-STEP SETUP FOR RASPBERRY PI:

*How to setup the **email2ncid** gateway on the Raspberry Pi for user **pi**.*

- *First go to the free dynamic IP service at https://www.changeip.com/dns.php and register a host name at their domain. For example: `foobar.freedynamicdns.us`*

- *Configure your firewall to pass port 25 TCP and UDP to the Raspberry Pi IP address.*
  The Raspberry Pi must have a fixed IP address or a static DHCP lease.

- *Install programs (mutt is recommended for a mail reader):*

```
sudo apt-get install procmail exim4 mutt
```

- *Configure exim4:*

```
sudo dpkg-reconfigure exim4-config
```

with the following settings:

Configuration Parameter	***Select or Type In***
Mail Server Configuration info screen	`Ok`
General type of mail configuration:	`mail sent by smarthost; received via SMTP or fetchmail`
System mail name: `raspberrypi`	*Accept default*
Mail Server Configuration info screen	`Ok`
IP-addresses to listen on for incoming SMTP connections: `127.0.0.1 ; ::1`	*Replace with* `0.0.0.0`
Other destinations for which mail is accepted: `raspberrypi`	*Add a semicolon and then your Internet host name:* `raspberrypi;foobar.freedynamicdns.us`
Machines to relay mail for:	*Leave blank*
IP address or host name of the outgoing smarthost:	*Change to blanks if no outgoing mail or enter your service provider outgoing smarthost*
Hide local mail name in outgoing mail?	`No`
Mail Server Configuration info screen	`Ok`
Keep number of DNS-queries minimal (Dial-on-Demand)?	`No`
Delivery method for local mail:	`mbox format in /var/mail/`
Split configuration into small files?	`No`
Root and postmaster mail recipient:	*Add user id, for Raspberry Pi it is usually* `pi`

- *Start exim4:*

```
sudo invoke-rc.d exim4 start
```

- *Enable exim4 at boot:*

```
sudo update-rc.d exim4 defaults
```

- *Run the following setup script to create or update $HOME/.procmailrc:*

```
ncid-setup email2ncid
```

- *If the NCID server is **not** running on the Raspberry Pi, edit file email2ncid.conf, uncomment the line for variable NCIDSERVER and set it to the correct IP address (or hostname) and port. For example:*

```
NCIDSERVER=192.168.10.55:3333
```

> - *Test with a 2 line mail message:*

```
mail pi@foobar.freedynamicdns.us
Subject: NCID Message
My first
email message.
```

> - *The resulting NCID message that is broadcast to all clients should be one line:* ***My first email message.***

ncid2ncid setup

> *How to setup the ncid2ncid gateway to forward caller ID and messages from multiple NCID sending servers (four maximum) to a single NCID receiving server.*

> ***Sections:***

> *REQUIREMENTS*
> *CONFIGURATION*
> *TESTING*
> *START/STOP/RESTART/STATUS/AUTOSTART*

> ***REQUIREMENTS:***

> *One NCID receiving server and at least one NCID sending server.*

> ***CONFIGURATION:***

> - *Receiving Server*

> *The ncid2ncid process connects to a receiving server as a gateway, that is, a device that is a source of caller ID and messages. Typically, ncid2ncid is running on the receiving server and for this reason the default receiving server is 127.0.0.1:3333. This can be changed by using the tohost and toport variables in file* ***ncid2ncid.conf****, or by using the* `-t|--tohost [host][:port]` *arguments on the command line.*

> *A receiving server is required.*

> - *Sending Servers*

> *The ncid2ncid process connects to sending servers as if it is a client. You specify sending servers using the fromhostX and fromportX variables in file* ***ncid2ncid.conf****, where X is a digit 1-4. You can also configure the sending servers from the command line by using multiple* `-f|--fromhost [host][:port]` *arguments.*

> *Note the subtle difference: Only specify a digit 1-4 for variables in file* ***ncid2ncid.conf****; do not use them on the command line. The following examples are equivalent:*

> ***ncid2ncid.conf****:*

```
    set fromhost1 = 192.168.20.1
    set fromport1 = 3334
    set fromhost2 = 192.168.20.9:3335
```

> *command line:*

```
ncid2ncid --fromhost 192.168.20.1:3334 --fromhost 192.168.20.9:3335
```

There are no default sending servers and at least one must be specified.

- *The default port for all sending/receiving servers is 3333.*

TESTING:

*Start **ncid2ncid** in debug mode at verbose level 3:*

```
sudo ncid2ncid -Dv3
```

*Debug mode will give a reason if **ncid2ncid** dies and will show its processing data.*

If ncid2ncid does not work, you should have enough information to ask for help.

START/STOP/RESTART/STATUS/AUTOSTART:

Normally ncid2ncid is started using the provided init, service, rc, or plist script for your OS. For more information, refer to the INSTALL section for your OS. If no script is provided you need to start ncid2ncid manually:

```
sudo ncid2ncid
```

*If any options are needed, add them to **ncid2ncid.conf**.*

obi2ncid setup

How to setup an Obihai device to send Caller ID and messages via the obi2ncid gateway.

Sections:

REQUIREMENTS:

An Obihai OBi device is required. Only the OBi100, OBi110, OBi200 with and without OBILINE and OBi202 were available for development and testing. Other OBIHAI products may or may not work.

*It is possible to configure multiple OBi devices to work with NCID. Each device must have a unique port configured and the DEVICE CONFIGURATION and NCID CONFIGURATION must use the same port. Ports 4335 through 4339 are normally used. Port 4339 is used by the **test-obi-gw** test script in the NCID source package.*

You must install your OBi device on your local network, create a free OBiTALK account and link your OBi to your account so that it appears on your OBiTALK Dashboard. Then you need to set up your voice service provider(s) (there are "wizards" that make it easy to set up the most common ones).

Make sure you can successfully make and receive calls before continuing.

DEVICE CONFIGURATION:

The OBi device needs to be configured for NCID use. You can do this using either the "Advanced Configuration" (a.k.a. "Expert") mode accessible through the OBiTALK dashboard, or by using a browser to login directly to the OBi device using its IP address. The OBiTALK Dashboard is the simplest and easiest method and is what will be used below.

On the OBiTALK dashboard, find your OBi device and click on the dark gray gear icon with the red "E" (for "Expert" or Advanced Configuration).

*Navigate to **System Management->Device Admin**.*

- *Find the **Syslog** section.*
- *Override the **Server** parameter by first UNCHECKING **OBiTALK Settings** and then immediately UNCHECK **Device Default**.*
- *Under the **Value** column for **Server**, type in the NCID server hostname or IP address.*
- *Override the **Port** parameter by first UNCHECKING **OBiTALK Settings** and then immediately UNCHECK **Device Default**.*
- *Under the **Value** column for **Port**, change the default of **514** to **4335**.*
- *Click on the **Submit** button to save the changes.*
- *The page will be redisplayed. At the top you should see the message Successfully saved OBi Expert config to OBiTALK or something similar.*

*Navigate to **Voice Service**.*

- *There will be a list that starts with "SP1 Service" and depending on the OBi model will go up to "SP4 Service." For any and all of these Services that is not configured for GTALK (a.k.a. Google Voice), scroll to where you see the parameter name of **X_SipDebugOption** (it is usually a few lines above the start of the **SIP Credentials** section).*
- *Override the **X_SipDebugOption** parameter by first UNCHECKING **OBiTALK Settings** and then immediately UNCHECK **Device Default**. Under the **Value** column, change the default of **Disable** to **Log All Except REGISTER Messages**.*
- *Click on the **Submit** button to save the changes.*
- *The page will be redisplayed. At the top you should see the message Successfully saved OBi Expert config to OBiTALK or something similar.*

NCID CONFIGURATION:

*The ncidd server defaults to using a modem and optional gateways to get Caller ID. If you are using a modem for Caller ID on a POTS (standard telephone) line, you can use the obi2ncid gateway to handle Caller ID from VoIP. No need to configure **ncidd.conf**.*

*If you are not using a modem for Caller ID, you need to configure ncidd by changing this line in **ncidd.conf**:*

```
From:       # set cidinput = 3
To:         set cidinput = 3
```

*If you are using a modem for hangup or to dial calls, you must have a modem connected to the same telephone line as the OBi device, and you must configure ncidd by changing these lines in **ncidd.conf**:*

```
From:       # set cidinput = 2
To:         set cidinput = 2
```

```
From:       # set lineid = POTS
To          set lineid = <OBi lineid>
```

Notes:

> *the **OBi lineid** depends on the service provider*
> *see Note 1 for an explanation of cidinput*

*Once you change **ncidd.conf**, you must start/restart ncidd to read it.*

Normally obi2ncid does not need to be configured unless you set up the OBi device to use a syslog port other than 4335.

EXAMPLE:

This ncidd.conf example is for a OBi device:

- *connected to the Google Voice service provider*
- *a modem connected to the same line input to the OBi device*
- *automatic plain hangup on unwanted calls:*

```
set cidinput = 2
set lineid = "GTALK"
```

TESTING:

If this is the first time you set obi2ncid up, you should test obi2ncid without connecting it to ncidd. For example, if using a obi200:

```
obi2ncid -t -L obi200.log
```

The above command puts obi2ncid in test mode at verbose level 3. It will display verbose statements on the terminal, ending with "Listening at port 4335". Test mode prevents obi2ncid from connecting with ncidd.

If obi2ncid terminates you should be able to see why and fix it.

If you have a problem that requires debugging you should use verbose level 5 and create a data file. For example, if using a obi200:

```
obi2ncid -t -v5 -L obi200.log -R obi200.data
```

You can get a detailed usage message by executing:

```
obi2ncid --help
```

or print out the manual page by executing:

```
obi2ncid --man
```

*If it looks OK, terminate obi2ncid with **<CTRL><C>**.*

Next, restart obi2ncid in debug mode so it will connect to ncidd:

```
obi2ncid -Dv3
```

*Make a call and you should see the **CALL** and **CALLINFO** lines that are sent to the server. You should also see **CID** and **END** lines sent back from the server. If there is a problem an error message may be generated.*

OUTPUT TESTING:

This is an advanced set of tests used when adding a new voice provider or device to the obi2ncid. It is also after fixing a problem with a voice provider, to make sure the fix did not break anything.

Suggested setup for easiest testing:

- *If testing both the OBi110 and OBi200, leave the OBi110 at port 4335 and change the OBi200 port to 4336.*
- *It's best to test at verbose 3 in test mode to just see **CALL** and **CALLINFO** lines:*

```
perl obi2ncid.pl -t -L test.log -C obi2ncid.conf -o 4335
```

- *If there is a problem with a test or tests, change to v5, then after each test move the test.log and test.data files to another name so there is one test call per log.*

```
perl obi2ncid.pl -t -L test.log -R test.data -C obi2ncid.conf \
                -o 4335 -v5
```

*Complete testing of the OBi110 and OBi200 requires checking that the name, number and line id are the same for the **CALL** and **CALLINFO** lines. In addition **CALLINFO** should show **CANCEL** if there was no pickup or **BYE** if there was a pickup.*

To verify all is working correctly you need to test the following:

```
Incoming call:
     originating caller hangup before answer
     originating caller hangup after answer
     receiving caller hangup after answer
```

```
Outgoing call:
     originating caller hangup before answer
     originating caller hangup after answer
     receiving caller hangup after answer
```

> *Additional tests for POTS calls. On the OBi110, connect the house Telco line to the OBi110 Telco Line connector. The OBi200 requires the OBiLINE FXO to USB Phone Line Adapter; connect the house Telco line to the OBiLINE.*

```
      Incoming Telco Line
           hangup with no answer
           hangup after house phone answers
           hangup after answer on the OBi device
```

```
      Outgoing Telco Line
           hangup with no answer
           hangup after house phone answers
           hangup after answer on the OBi device
```

> ***START/STOP/RESTART/STATUS/AUTOSTART:***

> *Normally obi2ncid is started using the provided init, service, rc, or plist script for your OS. For more information, refer to the INSTALL section for your OS. If none is provided you need to start obi2ncid manually:*

```
    sudo obi2ncid &
```

> *You can also set it up to start at boot, along with ncidd. If any options are needed, add them to* ***obi2ncid.conf****.*

> *If obi2ncid does not work, you should have enough information to ask for help.*

rn2ncid setup

> *How to setup Remote Notifier on an Android smart phone to send Caller ID and messages via the rn2ncid gateway.*

> ***Sections:***

> *REQUIREMENTS*
> *CONFIGURATION*
> *TESTING*
> *START/STOP/RESTART/STATUS/AUTOSTART*

> ***REQUIREMENTS:***

> *The smart phone needs to be running Remote Notifier for Android from F-Droid.*

> *Install it and configure it for the IP address of the computer running ncidd.*

> ***CONFIGURATION:***

> *The ncidd server defaults to using a modem and optional gateways to get Caller ID. If you are using a modem for Caller ID on a POTS (standard telephone) line, you can use the rn2ncid gateway to handle Caller ID from smart phones. No need to configure* ***ncidd.conf****.*

> *If you are not using a modem or serial device for Caller ID, you need to configure ncidd by changing this line in* ***ncidd.conf****:*

```
From:       # set cidinput = 3
To:         set cidinput = 3
```

*Hangup is not supported for Caller ID from the **rn2ncid** gateway.*

*Once you change **ncidd.conf**, you must start/restart ncidd to read it.*

(Note: See Note 1 for an explanation of cidinput)

*Normally rn2ncid does not need to be configured unless you are using ncid-page to send calls and messages to your smart phone. In that case you need to edit the reject line at the end of **rn2ncid.conf** and specify the "from" of SMS/MMS messages to be rejected and not passed through to the NCID server. (If you do not do this, the result will be an endless loop which could result in excessively high data or text charges by your cell phone carrier.) The setting for reject is usually of the form root@[hostname] where [hostname] is the result of executing the Unix `hostname` command on the computer running ncidd.*

For example:

> ***$** hostname*
> *smurfzoo.private*
> ***$***

*Edit **rn2ncid.conf**:*

```
reject = root@smurfzoo.private
```

TESTING:

If this is the first time you set rn2ncid up, you should test rn2ncid without connecting it to ncidd.

```
rn2ncid --test
```

The above command puts rn2ncid in test and debug modes at verbose level 3. It will display verbose statements on the terminal, ending with "Listening at port 10600". It should show configured options. Test mode prevents rn2ncid from connecting with ncidd.

If rn2ncid terminates you should be able to see why and fix it.

You can get a detailed usage message by executing:

```
rn2ncid --help
```

or print out the manual page by executing:

```
rn2ncid --man
```

On your smart phone, launch Remote Notifier and choose "Send test notification". rn2ncid should show something like this:

```
NOT: PHONE 0123: PING Test notification
```

(Note: 0123 is the phone ID and will be different for your phone.)

*If it looks OK, terminate rn2ncid with **<CTRL><C>**.*

> *Next, restart rn2ncid in debug mode so it will connect to ncidd:*

```
    rn2ncid -Dv3
```

> *Do the "Send test notification" again and it should be sent to the server and its clients. If you do not get a "NOT" (short for "NOTIFY") message sent to the server, you should instead get an error message saying what is wrong.*

> *If you had the PING "Test notification" sent to a client, setup is complete.*

> ***START/STOP/RESTART/STATUS/AUTOSTART:***

> *Normally rn2ncid is started using the provided init, service, rc, or plist script for your OS. For more information, refer to the INSTALL section for your OS. If none is provided you need to start rn2ncid manually:*

```
    sudo rn2ncid &
```

> *You can also set it up to start at boot, along with ncidd. If any options are needed, add them to* ***rn2ncid.conf****.*

> *If rn2ncid does not work, you should have enough information to ask for help.*

sip2ncid setup

> *How to setup VoIP Caller ID using sip2ncid.*

Sections:

> ***REQUIREMENTS:***

> *VoIP telephone service using SIP.*

> *The sip2ncid gateway automatically monitors SIP packets that use either TCP or UDP protocols.*

> *Configure your LAN for SIP. See ATA (Analog Terminal Adapter).*

> *Special requirements for sip2ncid under Cygwin:*

- *Download latest WinPcap installer from https://www.winpcap.org/*
- *Run the WinPcap installer and allow to start on boot*

- *WpdPack*

> *The last test by NCID Developers was using `WinPcap_4_1_3.exe` and `WpdPack_4_1_2.zip`.*

> ***Install WinPcap from windows if using the sip2ncid gateway:***

- *if compiling ncid from source you also need to:*

> - *download the latest WpdPack zip file*
> - *unzip the WpdPack zip file to create the WpdPack directory*

> ***CONFIGURATION:***

> *The ncidd server defaults to using a modem and optional gateways to get Caller ID. If you are using a modem for Caller ID on a POTS (standard telephone) line, you can use the sip2ncid gateway to handle Caller ID from VoIP. No need to configure* ***ncidd.conf****.*

> *If you are not using a modem or serial device for Caller ID, you need to configure ncidd by changing this line in* ***ncidd.conf****:*

```
From:       # set cidinput = 3
To:         set cidinput = 3
```

> *If you are using a modem for hangup or to dial calls, you must have a modem connected to the same telephone line as the telephone used to receive and make SIP calls, and you must configure ncidd by changing these lines in* ***ncidd.conf****:*

```
From:       # set cidinput = 2
To:         set cidinput = 2
```

```
From:       # set lineid = POTS
To          set lineid = <SIP lineid>
```

> *Notes:*

> > *the* ***SIP lineid*** *depends on the telephone line*
> > *see Note 1 for an explanation of cidinput*

> *Once you change* ***ncidd.conf****, you must start/restart ncidd to read it.*

> *Use the sip2ncid* `--listdevs` *or* `-l` *option to see your network devices:*

```
sudo sip2ncid --listdevs
```

> ***EXAMPLE:***

> *This ncidd.conf example is for SIP:*

> - *configured for phone number 786-555-1212*
> - *a modem connected to the same phone line as the telephone*
> - *automatic plane hangup on unwanted calls:*

```
set cidinput = 2
set lineid = "1212"
```

> ***TESTING:***

> *To determine if you can receive any network packets, use the* `--testall` *or* `-T` *option:*

```
sudo sip2ncid --testall
```

This will display a packet count and a packet type. It does not know all packet types so you may get some UNKNOWN packet types. It also sets debug mode and verbose level 3. You can increase the verbose level to see more detail, but if you decrease it below 3, you will not see any packets.

To determine if you can receive SIP data packets, use the `--testsip` or `-t` option:

```
sudo sip2ncid --testsip
```

This will display SIP packets and what, if anything, it does. It also sets debug mode and verbose level 3. You can also change the verbose level. If you set verbose to 1, sip2ncid will display lines sent to the server instead of the packet contents:

```
sudo sip2ncid -tv1
```

If no packets are received in about 45 seconds:

```
No SIP packets at port XXXX in XX seconds
```

If sip2ncid terminates you should be able to see why and fix it.

You can get a detailed usage message by executing:

```
sip2ncid --help
```

Sometimes it picks the wrong default interface. If you are using eth0:

```
sudo sip2ncid -ti eth0
```

If you need to see what interfaces are present you can use the `--interface` or `-i` option:

```
sudo sip2ncid --listdevs
```

The display is:

```
<interface> : <description>
```

The interface name is everything up to the first space.

If you do not see any SIP packets, change to port 5061 and try again:

```
sudo sip2ncid --testsip --sip :5061
```

You should see something like:

```
Network Interface: eth0
Filter: port 10000
```

Then about every 20 seconds you should see something like:

```
Packet number 1:
Protocol: UDP
```

```
     SIP/2.0 200 OK
     Via: SIP/2.0/UDP 70.119.157.214:10000;branch=z9hG4bK-22b185d1
     From: 321-555-7722
<sip:13215551212@atlas2.atlas.vonage.net:10000>;tag=46f26356c0a3394bo0
     To: 321-555-7722 <sip:13215551212@atlas2.atlas.vonage.net:10000>
     Call-ID: fa72d1c2-ead1bdcf@70.119.157.214
     CSeq: 86785 REGISTER
     Contact: 321-555-1212 <sip:13215551212@70.119.157.214:10000>;expires=20
     Content-Length: 0

     Registered Line Number: 13215551212
```

> *The Registered Line Number line will only appear in packet number 1.*

> *If you do not get the above, you may need to specify an address and/or port for sip2ncid to listen for the SIP Invite. You cannot continue unless you get the above.*

> *If you are using the Linksys RT31P2 Router, you will not see any packets unless the computer is in its DMZ (Demilitarized Zone). Port forwarding the UDP port will not work. You must set up the DMZ. If you are using a different VoIP router, try to put the computer in the DMZ and see if that works. If not, view the SIP tutorial:*

> *Configure your home network for SIP-based Caller ID (on the Wayback Machine).*

> *Once you receive the above packets, call yourself. If you do not get a Caller ID message sent to ncidd, you should get an error message saying what is wrong. This has been tested with Vonage and may need tweaking for other VoIP service providers.*

> *If you had Caller ID sent to a client, setup is complete. You can then restart sip2ncid without the test option so it will not display anything. You can also set it up to start at boot, along with ncidd. If any options are needed at boot, add them to **sip2ncid.conf**.*

> ***START/STOP/RESTART/STATUS/AUTOSTART:***

> *Normally sip2ncid is started using the provided init, service, rc, or plist script for your OS. For more information, refer to the INSTALL section for your OS. If none is provided you need to start sip2ncid manually:*

```
     sudo sip2ncid &
```

> *You can also set it up to start at boot, along with ncidd. If any options are needed, add them to **sip2ncid.conf**.*

> *If sip2ncid does not work, you should have enough information to ask for help.*

wc2ncid setup

> *How to setup one or more Whozz Calling Ethernet Link (WC) devices for Caller ID using wc2ncid.*

> ***Sections:***

> > *REQUIREMENTS*
> > *CONFIGURATION*
> > *EXAMPLE*
> > *TESTING*
> > *START/STOP/RESTART/STATUS/AUTOSTART*

> ***REQUIREMENTS:***

> *A Whozz Calling Ethernet Link (WC) device (see: https://callerid.com) connects to POTS (Plain Old Telephone System) lines and can handle 2, 4, or 8 lines. Some models only handle incoming calls while others handle incoming and outgoing calls.*

> *The Whozz Calling user manual tells how to hook up the device. You plug your POTS telephone lines into the device and you connect the device to your local network.*

> ***CONFIGURATION:***

> *All WC devices must have an IP address within your network in order for them to be configured for use by wc2ncid. This limitation will be removed in a future release. When you try to configure a device with an address outside your network, wc2ncid will either give a warning or an error message and terminate. You can then use the wct script to change the IP address to one that is in your network. Use the discover option of wct to locate the device:*

```
wct --discover
```

> *The ncidd server defaults to using a modem and optional gateways to get Caller ID. If you are using a modem for Caller ID on a POTS (standard telephone) line, you can use the wc2ncid gateway to handle Caller ID from additional POTS or VoIP lines. No need to configure* ***ncidd.conf****.*

> *If you are not using a modem or serial device for Caller ID, you need to configure ncidd by changing this line in* ***ncidd.conf****:*

```
From:       # set cidinput = 3
To:         set cidinput = 3
```

> *If you are using a modem for hangup or to dial calls, you must have a modem connected to the same telephone line as the WC device, and you must configure ncidd by changing these lines in* ***ncidd.conf****:*

```
From:       # set cidinput = 2
To:         set cidinput = 2
```

```
From:       # set lineid = POTS
To          set lineid = <WC lineid>
```

> *Notes:*

>> *the* ***WC lineid*** *is WC followed by 2 digits indicating the line input to the device*
>> *see Note 1 for an explanation of cidinput*

> *Once you change* ***ncidd.conf****, you must start/restart ncidd to read it.*

> *Next, edit* ***wc2ncid.conf*** *to configure one or more devices. Look for this line:*

```
wcaddr = 192.168.0.90
```

> *If your network is on 192.168.0 and the above address is not used, you can leave it. If your network is on 192.168.1 you can set the IP address for WC device number 1 (WC-1), for example, by changing the line to be:*

```
wcaddr = 192.168.1.90
```

If you have 2 devices and want to use addresses 192.168.2.90 and 192.168.2.91, WC device 1 is 192.168.2.90 and WC device 2 is 192.168.2.91.

```
    wcaddr = 192.168.2.90, 192.168.2.91
```

EXAMPLE:

This ncidd.conf example is for a WC device:

- *with a phone line connected to input 01*
- *a modem connected to the same line input to the WC device*
- *automatic plain hangup on unwanted calls*

```
    set cidinput = 2
    set lineid = "WC01"
```

TESTING:

*Once you set the IP address for the WC device in **wc2ncid.conf**, start wc2ncid and tell it to configure the WC device:*

```
    wc2ncid [--test] --set-wc
```

The `--test` parameter is optional, but it is a good idea to use it so wc2ncid does not connect to the NCID server during the configuration process.

*If you have 2 or more WC devices and they are both set to the same address or the factory default of 192.168.0.90, you need to change both addresses in **wc2ncid.conf**. For example:*

```
    wcaddr = 192.168.0.91, 192.168.0.92
```

Turn on one device and execute:

```
    wc2ncid [--test] --set-wc
```

*Terminate wc2ncid with **<CTRL><C>**. Leave the first device turned on, then turn on the second device and execute:*

```
    wc2ncid [--test] --set-wc
```

*Both devices should be configured and operational. Terminate wc2ncid with **<CTRL><C>**.*

If this is the first time you set wc2ncid up, you should test wc2ncid without connecting it to the ncidd server:

```
    wc2ncid --test
```

The above command puts wc2ncid in test and debug modes at verbose level 3. It will display verbose statements on the terminal, ending with "Waiting for calls". It should show the configured address for each device. Test mode prevents wc2ncid from connecting with ncidd.

> *If wc2ncid terminates you should be able to see why and fix it.*

> *You can get a detailed usage message by executing:*

```
wc2ncid --help
```

> *or print out the manual page by executing:*

```
wc2ncid --man
```

> *Call yourself. You should see more verbose messages as the call is processed. If it looks OK, terminate wc2ncid with* ***<CTRL><C>****.*

> *Next, restart wc2ncid in debug mode so it will connect to ncidd:*

```
wc2ncid -Dv3
```

> *Call yourself. If you do not get a Caller ID message sent to ncidd, you should get an error message saying what is wrong.*

> *If you had Caller ID sent to a client, setup is complete.*

> ***START/STOP/RESTART/STATUS/AUTOSTART:***

> *Normally wc2ncid is started using the provided init, service, rc, or plist script for your OS. For more information, refer to the INSTALL section for your OS. If none is provided you need to start wc2ncid manually:*

```
sudo wc2ncid &
```

> *You can also set it up to start at boot, along with ncidd. If any options are needed, add them to* ***wc2ncid.conf****.*

> *If wc2ncid does not work, you should have enough information to ask for help.*

xdmf2ncid setup

> *How to setup one or more CTI Comet USB devices or Holtek HT9032D based PSTN Caller ID modules for Caller ID using xdmf2ncid.*

> ***Sections:***

> *REQUIREMENTS*
> *CONFIGURATION*
> *EXAMPLE*
> *TESTING*
> *START/STOP/RESTART/STATUS/AUTOSTART*

> ***REQUIREMENTS:***

> *The CTI Comet USB device and the Holtek HT9032D based PSTN Caller ID module connect to a POTS (Plain Old Telephone System) lines to provide Caller ID on incoming calls.*

> ***CONFIGURATION:***

> *Connect the CTI Comet USB to the computer using the supplied USB cable or connect the Holtek HT9032D based PSTN Caller ID module to the computer using the USB to UART TTL cable adapter and then connect the CTI Comet USB or the Holtek HT9032D based PSTN Caller ID module to the telephone line using the remaining cable.*

> *The ncidd server defaults to using a modem and optional gateways to get Caller ID. If you are using a modem for Caller ID on a POTS (standard telephone) line, you can use the xdmf2ncid gateway to handle an additional POTS or VoIP line. No need to configure* ***ncidd.conf****.*

> *If you are not using a modem or serial device for Caller ID, you need to configure ncidd by changing this line in* ***ncidd.conf****:*

```
From:       # set cidinput = 3
To:         set cidinput = 3
```

> *If you are using a modem for hangup or to dial calls, you must have a modem connected to the same telephone line as the device connected to xdmf2ncid, and you must configure ncidd by changing these lines in* ***ncidd.conf****:*

```
From:       # set cidinput = 2
To:         set cidinput = 2
```

```
From:       # set lineid = POTS
To          set lineid = <device name>
```

> *Notes:*

- *the* ***xdmf2ncid lineid*** *is the device name*
- *see Note 1 for an explanation of cidinput*

> *Once you change* ***ncidd.conf****, you must start/restart ncidd to read it.*

> *Next, edit* ***xdmf2ncid.conf*** *to configure the USB port used by the CTI Comet USB or the Holtek HT9032D based PSTN Caller ID module. Uncomment the line and change the USB port. For Linux systems, ttyUSB? is normally used where USB? is USB0, USB1, USB2, or USB3.*

> *For example, to configure for USB1 uncomment this line:*

```
From:       #usbport = /dev/ttyUSB1
To:         usbport = /dev/ttyUSB1
```

> *You may need to change other settings in* ***xdmf2ncid.conf****. For instance, the configuration assumes ncidd is running on the same computer using its default port. If using the Holtek HT9032D based PSTN Caller ID module, set* ***ht9032 = 1****.*

> ***EXAMPLE:***

> *This ncidd.conf example is for the CTI Comet USB or the Holtek HT9032D based PSTN Caller ID module:*

- *connected to /dev/CometUSB0 or /dev/HoltekUSB0*
- *a modem connected to the same phone line connected to the CTI Comet USB device or the Holtek HT9032D based PSTN Caller ID module*
- *automatic plain hangup on unwanted calls*

```
# CTI Comet USB:
set cidinput = 2
set lineid = "CometUSB0"
```

```
# Holtek HT9032D based PSTN Caller ID module:
set cidinput = 2
set lineid = "HoltekUSB0"
```

TESTING:

Once you set the USB port for the CTI Comet USB device or the Holtek HT9032D based PSTN Caller ID module in ***xdmf2ncid.conf****, start xdmf2ncid:*

```
xdmf2ncid [--test]
```

The `--test` *parameter is optional, but it is a good idea to use it so that xdmf2ncid does not connect to the NCID server during the configuration process. You'll want to use the parameter if this is the first time you are setting xdmf2ncid up:*

```
xdmf2ncid --test
```

The above command puts xdmf2ncid in test and debug modes at verbose level 3. It will display verbose statements on the terminal, ending with "Waiting for calls". It should show the USB port for the device.

If xdmf2ncid terminates you should be able to see why and fix it.

You can get a detailed usage message by executing:

```
xdmf2ncid --help
```

or print out the manual page by executing:

```
xdmf2ncid --man
```

Call yourself. You should see more verbose messages as the call is processed. If it looks OK, terminate xdmf2ncid with ***<CTRL><C>****.*

Next, restart xdmf2ncid in debug mode so it will connect to ncidd:

```
xdmf2ncid -Dv3
```

Call yourself. If you do not get a Caller ID message sent to ncidd, you should get an error message saying what is wrong.

If you had Caller ID sent to a client, setup is complete.

START/STOP/RESTART/STATUS/AUTOSTART:

Normally xdmf2ncid is started using the provided init, service, rc, or plist script for your OS. For more information, refer to the INSTALL section for your OS. If none is provided you need to start xdmf2ncid manually:

```
sudo xdmf2ncid &
```

> *You can also set it up to start at boot, along with ncidd. If any options are needed, add them to **xdmf2ncid.conf**.*

> *If xdmf2ncid does not work, you should have enough information to ask for help.*

yac2ncid setup

> *How to setup a YAC modem server for Caller ID using yac2ncid.*

> ***Sections:***

> *REQUIREMENTS*
> *CONFIGURATION*
> *TESTING*
> *START/STOP/RESTART/STATUS/AUTOSTART*

> ***REQUIREMENTS:***

> *A YAC server on a Windows computer running Microsoft Windows 98 or later.*

> *The YAC server has not been updated since approximately 2002. You can download it from a saved site copy at the Internet Archive's Wayback Machine, or from any of several global download sites such as https://yac.software.informer.com/. Follow the installation instructions.*

> ***CONFIGURATION:***

> *Configure the YAC server by giving it the IP address where ncidd is running. Do this by right-clicking the YAC icon in the System Tray and then select "Listeners...".*

> *The ncidd server defaults to using a modem and optional gateways to get Caller ID. If you are using a modem for Caller ID on a POTS (standard telephone) line, you can use the yac2ncid gateway to handle an additional POTS line. No need to configure **ncidd.conf**.*

> *If you do not use a modem or serial device for Caller ID, you need to configure ncidd by changing this line in **ncidd.conf**:*

```
From:       # set cidinput = 3
To:         set cidinput = 3
```

> *Hangup is not supported for Caller ID from the **yac2ncid** gateway.*

> *Once you change **ncidd.conf**, you must start/restart ncidd to read it.*

> *(Note: See Note 1 for an explanation of cidinput)*

> *Normally yac2ncid does not need to be configured, but you should review **yac2ncid.conf** to see if you want to change any of its defaults.*

> *After modifying **ncidd.conf** and **yac2ncid.conf**, you must start/restart ncidd first and then the yac2ncid gateway.*

> ***TESTING:***

> *Make sure the YAC server is running on the Windows computer.*

Run the yac2ncid gateway with the verbose option:

```
yac2ncid -v
```

Call yourself. If you do not get a Caller ID message sent to ncidd, you should get an error message saying what is wrong.

If you had Caller ID sent to a client, setup is complete. You can then restart yac2ncid without the verbose option so it will not display anything. You can also set it up to start at boot, along with ncidd.

START/STOP/RESTART/STATUS/AUTOSTART:

Normally yac2ncid is started using the provided init, service, rc, or plist script for your OS. For more information, refer to the INSTALL section for your OS. If none is provided you need to start yac2ncid manually:

```
sudo yac2ncid &
```

You can also set it up to start at boot, along with ncidd. If any options are needed, add them to ***yac2ncid.conf****.*

If yac2ncid does not work, you should have enough information to ask for help.

Note 1:

In ***ncidd.conf*** *cidinput specifies the Caller ID input:*

cidinput value	*description*
0	*Caller ID from a modem and optional gateways*
1	*Caller ID from a serial or USB device and optional gateways*
2	*Caller ID from a gateway with modem support*
3	*Caller ID from gateways without modem support*

- *A list of serial devices working with NCID can be found in the Devices Supported section.*

Supported Clients

Table of Contents

Clients Index

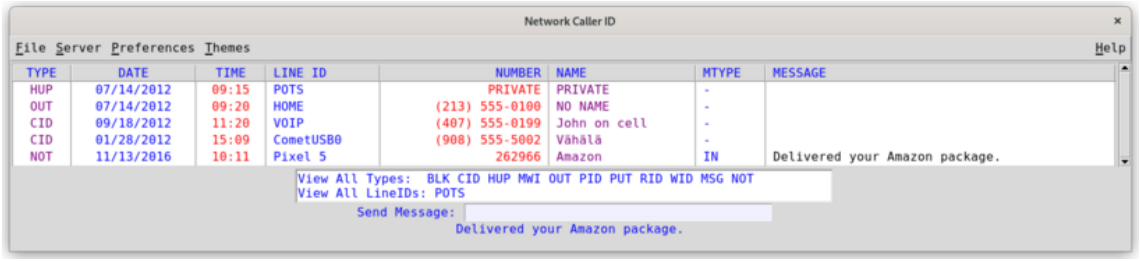
Overview

> *The primary NCID distribution includes one "universal" Graphical User Interface (GUI) client that runs under the Unix scripting language called Wish (Windowing Shell). In addition, there are four optional clients, available as separate downloads, all of which are actively maintained by the NCID Development Team.*

> *There is a [3rd Party Addons] page that links to clients, gateways and related NCID projects by other developers and users. Contact us with a URL showing your project and we will provide a link to it.*

> *The five clients are ncid, NCIDpop, NCID Android, LCDncid and NCIDdisplay.*

ncid



> *The `ncid` client is cross-platform for Linux, Mac OSX and Windows. It is included with the NCID distribution. The Windows version can only run the client without output modules.*

> *Description, Features and Screenshots*

> *Basic steps for starting the client:*

- *Make sure a window manager is running (e.g., X-Windows).*
- *Launch a graphical Terminal application (e.g., **xterm**).*
- *Make sure `ncidd` is running.*
- *To run `ncid` on the **same** computer as `ncidd`, at the Unix shell prompt, type in:*

```
ncid &
```

- *To run `ncid` on a **different** computer, specify the hostname or IP address of the `ncidd` computer:*

```
ncid 192.168.1.5 &
```

NCIDpop

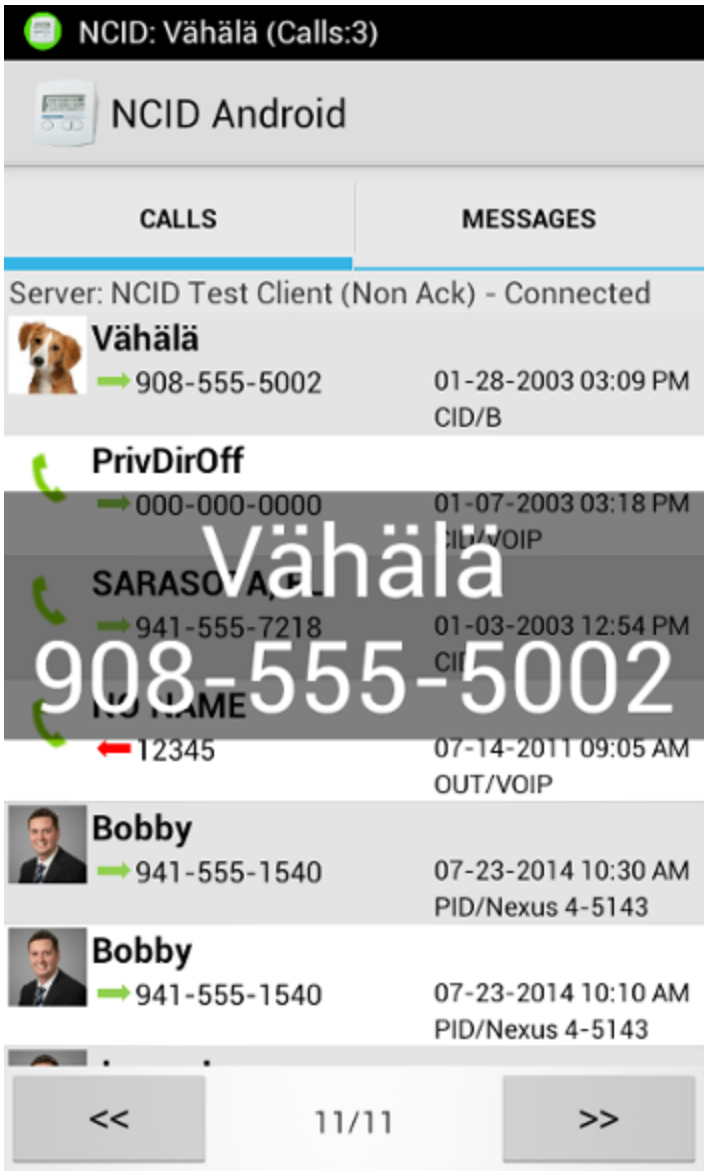
> *NCIDpop is an optional cross-platform popup client for Linux, Mac OSX and Windows.*

> *Description, Features and Screenshots*

> *Basic steps for starting the client:*

- *Click on the NCIDpop program icon.*
- *The first time NCIDpop runs it should take you to the General tab on the Preferences window.*
- *In the Server Settings box, type the hostname or IP address of the NCID server, e.g., 192.168.1.5.*
- *Click the Set button.*

NCID Android



NCID Android is an optional client for Android running on smartphones, tablets and Android TV devices.

Description, Features and Screenshots

Basic steps for starting the client:

- *Tap the NCID Android program icon.*
- *The first time NCID Android runs it should take you to the Preferences window.*
- *Tap Server Settings.*
- *Tap Hostname and type the hostname or IP address of the NCID server, e.g., 192.168.1.5.*
- *Tap the OK button.*

> - *Click on the BACK button until you get back to the screen showing Calls and Messages.*

LCDncid

> *LCDncid is an optional client that requires an LCD module connected to a Raspberry Pi or other computer.*

> *Description, Features and Screenshots*

> *Basic steps for starting the client:*

> - *You can run the NCID server and LCDncid client on the same device; the configuration file **lcdncid.conf** should have suitable defaults already.*

> - *If the LCDncid client is not running on the same device as the NCID server, you'll need to edit **lcdncid.conf** and set the ncidhost variable to the hostname or IP address, e.g.:*

```
ncidhost = 192.168.1.5
```

NCIDdisplay

> *NCIDdisplay is an optional giant (3"x18"!) homebrew LED display made up of at least 4 LCD modules.*

> *Description, Features and Screenshots*

> *Basic steps for starting the client:*

> - *NCIDdisplay is more involved than the other clients listed above because it requires the completion of its build-it-yourself hardware project.*
>
> - *When properly built, NCIDdisplay will run by simply powering it on.*

Client Output Modules

> *Table of Contents*
>
> *Description*
>
> *Usage*

Description

> *The ncid client supports output modules.*
>
> *An output module receives the Caller ID information from the ncid client and gives the client new functionality. For example, one module sends the Caller ID or message to a smart phone as an SMS message or sends it as an email message. Another module speaks the Caller ID or message.*
>
> *There are various output modules included with NCID. See the ncid_modules man page for a current list of the distributed modules. Each module is described in the man pages.*
>
> *There are also third party client modules. Some third party modules can be found at NCID Addons.*

Usage

> *Modules are called by the client using a command line of this form:*
>
> ```
> ncid --no-gui --module ncid-<name>
> ```
>
> *Most modules have a configuration file normally at /etc/ncid/conf.d:*
>
> ```
> /etc/ncid/conf.d/ncid-<name>.conf
> ```
>
> *As an example, if the <name> is **page** and you want to run the command in the background, then the command line would be:*
>
> ```
> ncid --no-gui --module ncid-page &
> ```
>
> *All distributed modules have a boot script that contains the proper command line to start and stop the module as well as giving its status. The boot scripts differ by operating systems. See the INSTALL document for more information on this for your operating system.*
>
> *The boot script name is always the same name as the module name. Use the AUTOSTART link for your operating system on how to start the module at boot. Use the START/STOP/RESTART/RELOAD/STATUS link for how to start and stop the module. Normally you would enable the module to start at boot after you configure it.*
>
> *Fedora: AUTOSTART START/STOP/RESTART/RELOAD/STATUS*
>
> *FreeBSD: AUTOSTART START/STOP/RESTART/RELOAD/STATUS*
>
> *Macintosh OSX: (AUTO)START/STOP*

> *raspios: AUTOSTART START/STOP/RESTART/RELOAD/STATUS*

> *Redhat: AUTOSTART START/STOP/RESTART/RELOAD/STATUS*

> *Ubuntu: AUTOSTART START/STOP/RESTART/RELOAD/STATUS*

Using NCID

> *Table of Contents*

> *Description*

Description

> *NCID's main job is to display the Caller ID on multiple devices, but it has advanced features.*

> *NCID can send and receive one line messages. It can also forward a message through a gateway. Review the **Messages** section for more information.*

> *NCID can give the caller a new name, called an alias. It can do the same with the caller's number or the line of the received call. Review the **Alias** section for more information.*

> *NCID can automatically hang up on the caller if the number or name is entered into its blacklist. Review the **Hangup** section for more information.*

> *NCID can execute a hangup extension as another way to determine if it should hangup on a call. Review the **Server Extensions** section for more information.*

> *NCID provides the ncidd.alias, ncidd.blacklist and ncidd.whitelist configuration files. The cidalias tool views alias definitions in the NCID alias, blacklist and whitelist files. The ncidutil tool modifies the ncidd.alias, ncidd.blacklist and ncidd.whitelist files. The cidupdate tool updates the current call log file using entries found in the alias file. Review the **Command Line Tools** section for more information.*

> *NCID provides a call log in the /var/log/cidcall.log file for calls and messages. It is processed each month if the operating system uses logrotate for the system log files. The cidcall tool is used to view the logfile.*

> *If configured, NCID also provides yearly call logs in the $HOME/NCID/log/cidcall-<year>.log files. Use the cidcall tool to view a yearly log file. Review the **Log Files** section for more information.*

> *Messages*
> *Aliases*
> *Hangup*
> *Server Extensions*
> *Command Line Tools*
> *CallerID Blacklist from FCC Data*
> *Enforcing the North American Numbering Plan*
> *Log Files*

NCID Messages

> *Table of Contents*

> *Description*

Description

> *NCID supports three different types of messages. All messages must be a single line. All messages must begin with either a MSG:, NOT:, or RLY: label.*

> ***MSG:***

> > *The server accepts a single line text message from a client and broadcasts it to all connected clients. All messages must begin with the MSG: label.*

> > *Gateways can send MSG: lines, too. For example, ncid2ncid will send MSG: lines to indicate the connection status as it passes caller ID data between two or more NCID servers.*

> > *Programs external to NCID, such as netcat (a.k.a. nc and ncat), can be used to send a message. Telnet is not recommended. If netcat is used, please note there are different versions with different options.*

> > *This shell script example creates a 10 minute food timer. The -w1 is a one second idle timeout to wait before disconnect:*

> > `sleep 600; echo "MSG: Food Ready" | nc -w1 localhost 3333 > /dev/null`

> > *An equivalent batch file using netcat for Windows would be:*

> > `CHOICE /T 600 /C AB /N /D A > NUL` `echo MSG: Food Ready | nc -w1 192.168.1.100 3333 > NUL`

> > > *Note: The recommended netcat version for Windows is the security-safe version by Jon Craton. It is available for download from his website here. When extracting, use the password "nc" (without quotes).*

> > *You can send a HELLO: CMD: no_log line prior to a MSG: line. This can improve performance because it forces the server not to send the call log before processing the MSG:.*

> > *Unix:*

> > `sleep 600; echo -e "HELLO: IDENT: client food timer 1.1\nHELLO: CMD: no_log\nMSG: Food Ready" | nc -w1 localhost 3333 > /dev/null`

> > *Windows batch file:*

> > `CHOICE /T 600 /C AB /N /D A > NUL` `(echo HELLO: IDENT: client food timer 1.1&echo HELLO: CMD: no_log&echo MSG: Food Ready) | nc -w1 192.168.1.100 3333 > NUL`

> ***NOT:***

> > *Utilizing the same format as MSG:, when a smartphone gateway (e.g., NCID Android) sends a copy of an SMS text message received or sent by a smartphone, it uses the NOT: line label.*

> ***RLY:***

> > *When a client (e.g., NCIDpop) sends a message to be forwarded over the cellular network, it uses the RLY: label to send it to a smartphone gateway (e.g., NCID Android). It uses a very different format compared to MSG: and NOT:.*

Aliases

> *Table of Contents*

Alias Index

Alias Overview

> *The name, number and telephone line of a call are checked for an alias. If a match is found it will be replaced by its alias before the call is added to the call log and before the call information is sent to the clients.*

> *There are two types of aliases;* ***built-in alias*** *and* ***user defined alias****.*

> *The built-in alias only converts the letter A to ANONYMOUS, the letter O to OUT_OF_AREA and the letter P to PRIVATE. The built-in aliases can not be modified but a built-in alias can be changed with a user defined alias.*

> *The user defined aliases are defined in the* ***ncidd.alias*** *file.*

> *A leading # is allowed in a name or number provided it is enclosed in double quotes. Sometimes a spoofed number starts with one or more # to prevent it from being blacklisted.*

> *Alias names can be a maximum of 50 characters.*

> *You can manually edit the* ***ncidd.alias*** *file to create, edit, or remove entries using an editor.*

> *When the alias file is modified,* ***ncidd*** *needs to be informed. It will reload the alias table when it receives a SIGHUP signal.*

> > *One way to send the reload signal to* ***ncidd****:*

> > > ***pkill --signal SIGHUP ncidd***

> > *You can also just restart* ***ncidd****.*

> *Aliases can also be created, edited, or removed by these NCID clients. They provide ways to force the server to reload the alias table.*

- *ncid*
- *NCIDpop*
- ~~*NCID Android*~~ *Alias maintenance not available*

> *In addition, when running the NCID server in the US or Canada, the* ***ignore1*** *option can be set in* ***ncidd.conf*** *to ignore a leading 1 in the Caller ID number or the alias number.*

Alias Types

> *The three general formats for entries in* ***ncidd.alias*** *are:*

- *alias "****received data****" = "****replacement data****"*

- *alias "**TYPE**" "**received data**" = "**replacement data**"*
- *alias "**TYPE**" "*****" = "**replacement data**" if "**depends**"*

Where:

- ***TYPE** is NAME, NMBR, or LINE*
- ***received data** or **depends** is the name or number to change to an alias*
- ***replacement data** is the alias*

*When **TYPE** is NAME and **depends** is present:*

- ***depends** is the received number to match against when setting the alias to be **replacement name**. The position normally used for **received data** must be an ****.*

*When **TYPE** is NMBR and **depends** is present:*

- ***depends** is the received name to match against when changing **received number** to **replacement number**. The position normally used for **received data** must be an ****.*

*When **TYPE** is missing, the NMBRNAME rule applies:*

- *if **received number** matches received data, it will be changed to **replacement data***

...and/or...

- *if **received name** matches received data, it will be changed to **replacement data***

The three general formats allow for six types of aliases:

Type	*Format*
number	*alias NMBR "received number" = "replacement number"*
name	*alias NAME "received name" = "replacement name"*
number & name	*alias "received data" = "replacement data"*
number if name	*alias NMBR = "*" = "replacement number" if "depends"*
name if number	*alias NAME = "*" = "replacement name" if "depends"*
lineid	*alias LINE = "received call lineid" = "replacement lineid"*

Number Alias

*The **number alias** changes the number of the caller to an alias. The number can be a word.*

*Format: **alias NMBR "received number" = "replacement number"***

*Example: **alias NMBR "4075550000" = "4075551212"***

Name Alias

*The **name alias** changes the name of the caller to an alias.*

*Format: **alias NAME "received name" = "replacement name"***

> *Example: alias NAME "John Bright" = "Big Bad John"*

> ***Number and Name Alias***

> *The **number and name alias** checks both the caller number and caller name for a match. If either matches, it is replaced by the alias.*

> *This alias type is useful when both the name and number are the same. For example, if the name and number are both "OUT-OF-AREA".*

> *Another use is if you do not care if the string you are looking for is a name or number. You want it replaced if it is either a name or number.*

> *Format: **alias "received data" = "replacement data"***

> *Example: **alias "OUT-OF-AREA" = "UNAVAILABLE"***

> ***Number if Name Alias***

> *The **number if name alias** changes the number of the caller if the name matches **depends**. Quotes around the "*****" are optional.*

> *Format **alias NMBR "*" = "replacement number" if "depends"***

> *Example: **alias NMBR * = "secret" if "Bond James"***

> ***Name if Number Alias***

> *The **name if number alias** is the most popular alias. It changes the name of the caller if the number matches **depends**. Quotes around the "*****" are optional.*

> *Format **alias NAME "*" = "replacement name" if "depends"***

> *Example: **alias NAME * = "john on cell" if "4075556767"***

> ***Line Alias***

> *The **line alias** changes the telephone line tag to an alias.*

> *Format **alias LINE "received call lineid" = "replacement lineid"***

> *Example: **alias LINE "-" = "POTS"***

Alias Expressions

> *Simple expressions (**set regex = 0** in **ncidd.conf**)*

> *These are permitted in either:*

> > *the **depends** section of an alias line*

> > *...or...*

> > *the **received data** section, if the alias line does not contain **depends***

> *The allowed expressions are:*

> - ***received data** and **depends** can contain a **^** or ***** or **1?** at the beginning:*

> > > *^ = partial match from beginning*
> > > ** = partial match after the **
> > > *^1? = optional leading 1 for US/Canada numbers*

> > - ***received data** and **depends** can contain an ***** at the end:*

> > > ** = partial match from beginning to before the **

> > - ***received data** and **depends** can contain an **$** at the end:*

> > > *$ = partial match tied to the end e.g. "* FL$" will match "MIAMI FL"*

> > - ***received data** can contain a single * to match anything*

> *Regular Expressions (**set regex = 1** in **ncidd.conf**)*
>
> > *They are used in place of Simple Expressions. The syntax for Simple Expressions is not compatible with Regular Expressions except for **^**, **^1?** and **1?**.*
>
> > *The POSIX Extended Regular Expression syntax is used. It is a section in Regular Expression.*
>
> > *If you are new to Regular Expressions, see Regular-Expressions info for an introduction.*

> *Perl-compatible Regular Expressions (**set regex = 2** in **ncidd.conf**)*
>
> > *They are used in place of Simple Expressions. The syntax for Simple Expressions is not compatible with Perl-compatible Regular Expressions except for **^**, **^1?** and **1?**.*
>
> > *This PCRE Regex Cheatsheet may be helpful.*

Hangup

> *Table of Contents*

Hangup Index

Hangup Overview

> *Hangup (a.k.a. call termination) is disabled/enabled by settings in the server configuration file* ***ncidd.conf****. It requires a modem to be connected to the phone line so it can pickup and hangup the line.*

> *At a high-level, there are two sets of procedures available to hangup calls. Both are optional and one or both can be enabled at the same time. They are:*

- *Internal Hangup. This is built in to the NCID server and uses the* ***ncidd.blacklist*** *and* ***ncidd.whitelist*** *files.*

- *Hangup Extension. This lets you use an external script or program. The* ***ncidd.whitelist*** *file is used to determine if the hangup script is called.*

> *When Caller ID is received from a modem, the following steps take place and in this order:*

- *The server looks for matching data in the optional alias file. This can result in the Caller ID name and/or number being changed.*

- *The server looks for matching data in the optional whitelist file. The data must be an alias, or if there is no alias for the call, it must be from the Caller ID. If there is a match then no hangup takes place.*

- *The server looks for matching data in the blacklist file. The data must be an alias, or if there is no alias for the call, it must be from the Caller ID. If there is a match, hangup is automatic.*

- *With the hupmode option, the server calls the external hangup extension. If the extension returns "hangup", hangup is automatic.*

> *You can manually edit the blacklist and whitelist files to create, edit, or remove entries using an editor.*

> *When either file is modified,* ***ncidd*** *needs to be informed. It will reload the blacklist and whitelist tables when it receives a SIGHUP signal.*

> *One way to send the reload signal to* ***ncidd****:*

```
sudo pkill --signal SIGHUP ncidd
```

> *You can also just restart* ***ncidd****.*

> *The blacklist and whitelist entries can also be created or removed by the following NCID clients. They provide ways to force the server to reload the tables:*

- *ncid*
- *NCIDpop*
- *NCID Android*

> *In addition, when running the NCID server in the US or Canada, the* ***ignore1*** *option can be set in* ***ncidd.conf*** *to ignore a leading 1 in the Caller ID, alias, blacklist, or whitelist.*

Modem Configuration

> *A modem is required to hangup the line. Hangup is enabled and configured in* ***ncidd.conf*** *by the* ***hangup*** *variable.*

> *Hangup is disabled by default, with and without a configuration file. Hangup can also be disabled by the line:*

> ***set hangup = 0***

> *You may also need to set the **ttyport** variable if the correct one is not set in **ncidd.conf**. For example, using Linux:*

> > ***set ttyport = /dev/ttyACM0***

> *The location of **ncidd.conf** will vary depending on the operating system. It is typically found in either **/etc/ncid/** or **/usr/local/etc/ncid/**. The **ncidd.conf** file location is also shown in **/var/log/ncidd.log** when **ncidd** starts. If **ncidd.conf** is missing, **ncidd.log** will have an appropriate message.*

> *Announce Hangup has some additional requirements for modem configuration.*

Hangup Choices

> *If enabled, there are three types of hangups:*

> - ***Normal Hangup** - requires a modem that supports Caller ID*
> - ***FAX Hangup** - requires a modem that supports Caller ID and FAX*
> - ***Announce Hangup** - requires a modem that supports Caller ID, VOICE and hardware flow control*

> *If the type of hangup is not supported by the modem, **ncidd** will change the hangup to **Normal Hangup**.*

> ***Normal Hangup***

> *A normal hangup is configured in **ncidd.conf** by the line:*

> > ***set hangup = 1***

> *When **ncidd** receives a blacklisted Caller ID, it will immediately hangup.*

> ***FAX Hangup***

> *A FAX hangup will generate a FAX tone for 10 seconds and then hangup. It is configured in **ncidd.conf** by the line:*

> > ***set hangup = 2***

> *Not all FAX modems are supported. If no FAX tones are generated, set **pickup** to 0 in **ncidd.conf**. It is usually needed for older modems.*

> > ***set pickup = 0***

> *After changing the pickup value, if it still does not work then the modem is not supported for **FAX Hangup** by **ncidd**.*

> ***Announce Hangup***

> *Announce Hangup will play a recorded message and then hangup.*

> *Configuration*
> *Voice Files*

> > *CX93001 Chipset Voice Files*
> > *US Robotics USR5637 Voice Files*
> > *Creating a Voice File from Scratch*

> ***Configuration***

Modem

Make sure the modem supports hardware flow control and that it is enabled. Look for the line ***Modem ACTIVE PROFILE:*** *in* ***ncidd.log****. For most modems, hardware flow control is indicated by the presence of &K3 in the profile. You may need to set &K3 in the active profile. If &K (or whatever is appropriate for your modem) is not returned, then the modem is not supported by NCID.*

ncidd.conf

Announce Hangup is configured by the line:

set hangup = 3

In addition, two other variables, ***announce*** *and* ***audiofmt****, are used to configure the announcement file. For example:*

set announce = DisconnectedNotInService.rmd set audiofmt = "AT+VSM=130"

If the announcement file is missing, ***ncidd*** *will change* ***Announce Hangup*** *to* ***Normal Hangup****. This will be indicated in* ***ncidd.log****.*

The default voice file is for 8-bit unsigned PCM at an 8000 Hz sample rate. It also seems to work for 8-bit LINEAR at an 8000 Hz sample rate.

The ***audiofmt*** *variable determines the voice file's compression method and sampling rate. The shorthand for this is* ***Voice Sampling Method*** *or VSM.*

Voice Files

CX93001 Chipset Voice Files

The default voice file supplied, ***DisconnectedNotInService.rmd****, works for modems that use the* ***CX93001*** *chipset. See the Incomplete list of working modems for known modems that use this chipset.*

Variable ***audiofmt*** *defaults to* ***AT+VSM=130*** *to work with this chipset.*

US Robotics USR5637 Voice Files

The default voice file will also work with the US Robotics USR5637 modem, but it must have firmware 1.2.23 or newer and ***audiofmt*** *must be changed.*

To check the firmware level, examine ***ncidd.log*** *for this line:*

Modem Identifier: U.S. Robotics 56K FAX USB V1.2.23

If you need to upgrade the firmware, download it from the US Robotics USR5637 support page.

The VSM line used is

128,"8-BIT LINEAR",(7200,8000,11025)

It has () around the supported voice sampling rates with three choices. You need to select 8000 for use with the default ***DisconnectedNotInService.rmd*** *file. Make the following change in* ***ncidd.conf****:*

set audiofmt = "AT+VSM=128,8000"

Creating a Voice File from Scratch

> > *This is a rather lengthy procedure so we have dedicated Hangup Appendix B: Creating a Voice File from Scratch to this topic.*

Blacklist and Alias Files Usage

> *File **ncidd.blacklist** is a list of names and/or numbers that will be terminated using one of the Hangup Choices above. By making use of expressions and wildcards, you can achieve more complex matching logic with even fewer entries in the blacklist. See the the ncidd.blacklist.5 man page for more info.*

> *Comments are supported and must begin with a **#**. Comments can be an entire line, or a comment can be at the end of an entry line.*

> *A leading # is allowed in a name or number provided it is enclosed in double quotes. Sometimes a spoofed number starts with one or more # to prevent it from being blacklisted.*

> *Beginning with NCID version 1.3, a **match name** is a special kind of comment that replaces the caller name or alias when it matches an entry in either the blacklist or whitelist. It can only be at the end of an entry line and must begin with **#=**.*

> *All phone numbers below are intended to be fictitious.*

- *Method 1: Using only blacklist file entries*

ncidd.blacklist	***Optional Comment***
2125550163	*# Free home security system*
2265196565	
2145559648	*# Win a free cruise*
3402090504	
8772954057	*# Acme Market Research*
9792201894	*# Political survey*
"#########8"	*# The quotes are required*

- *Method 2: Using multiple alias entries and one blacklist entry*

> *When a call comes in, NCID will apply any alias transformations before checking the blacklist file. You can use this to your advantage to simplify the number of entries needed in **ncidd.blacklist**. Another advantage this gives you is that NCID clients (e.g., NCIDpop) will show the caller name as TELEMARKETER to indicate the reason for the hangup.*

ncidd.alias	***Matching phone#***
*alias NAME * = "TELEMARKETER"*	*if 2125550163*
*alias NAME * = "TELEMARKETER"*	*if 2265196565*
*alias NAME * = "TELEMARKETER"*	*if 2145559648*
*alias NAME * = "TELEMARKETER"*	*if 3402090504*

*alias NAME * = "TELEMARKETER"*	*if 8772954057*
*alias NAME * = "TELEMARKETER"*	*if 9792201894*

ncidd.blacklist
"TELEMARKETER"

- *Method 3: Using only blacklist file entries with a **match name***

*Beginning with NCID version 1.3, Method 2 can be simplified further by using the blacklist line comment field to be a match name. This behaves like an alias but you don't need to make entries in **ncidd.alias**. The trick is to use the two characters **#=** as a special comment line. Spaces are optional between **#=** and the match name.*

*The blacklist match name overrides the incoming caller ID name and any **ncidd.alias** match that might exist.*

ncidd.blacklist	***Match Name in Comment***
2125550163	*#= Free home security system*
2265196565	*#= TELEMARKETER*
2145559648	*#= Win a free cruise*
3402090504	*#= TELEMARKETER*
8772954057	*#= Acme Market Research*
9792201894	*#= Political survey*
"##########8"	*#= Tried to avoid the blacklist*

Whitelist File Usage

*File **ncidd.whitelist** is a list of names and/or numbers that will prevent a blacklist entry from causing a hangup. By making use of expressions and wildcards, you can achieve more complex matching logic with even fewer entries in the whitelist. See the ncidd.whitelist.5 man page for more info.*

*Comments are supported and must begin with a **#**. Comments can be an entire line, or a comment can be at the end of an entry line.*

A leading # is allowed in a name or number provided it is enclosed in double quotes, just like in the blacklist.

*The whitelist match name is entered the same way as a blacklist match name, using **#=**.*

*The whitelist match name overrides the incoming Caller ID name and any **ncidd.alias** match that might exist.*

This example shows how to blacklist an entire area code while allowing specific numbers. It also shows how to indicate when there is a match in the whitelist.

ncidd.blacklist	Match Name in Comment
^999	#= Blacklist area code 999
^998	#= Blacklist area code 998

ncidd.whitelist	Match Name in Comment
9995556732	#= WHT 999-555-6732
9985550000	#= WHT James Bond

Alias, Blacklist and Whitelist Simple Expressions

The Blacklist and Whitelist can have either Simple Expressions or Regular Expressions (but not both) for an entry.

*In addition, when running the NCID server in the US or Canada, the **ignore1** option can be set in **ncidd.conf** to ignore a leading 1 in the Caller ID, alias, blacklist, or whitelist.*

*Simple Expressions (**set regex = 0** in **ncidd.conf**):*

> *^ = partial match from beginning*
> *1? = optional leading 1 (for US/Canada numbers)*

Alias, Blacklist and Whitelist Regular Expressions

*Regular Expressions are used in place of Simple Expressions. The syntax for Simple Expressions is not compatible with Regular Expressions except for **^**, **^1?** and **1?**.*

If you are new to Regular Expressions, see Regular-Expressions.info for an introduction or a Regular Expressions Tutorial.

*Posix Regular Expressions (**set regex = 1** in **ncidd.conf**):*

- *The POSIX Extended Regular Expression syntax is used.*
  It is a section in Regular Expression.

You can also refer to the Quick-Start: Regex Cheat Sheet

*Perl-compatible Regular Expressions (**set regex = 2** in **ncidd.conf**):*

- *The Perl-compatible regular expression syntax is used.*

You can also refer to the PCRE Regex Cheatsheet

Hangup Appendix A: Server Log File Sample

*The **ncidd.log** is useful for indicating **ncidd** configuration settings, modem features, modem settings and debugging information.*

*This is a portion of the output for a tty port and an LB LINK USB Modem. The output is at the default verbose 1 level. It is useful for reviewing the tty port parameters, the modem identifier, country code, **Voice Sampling Methods** supported, the VSM selected and the setting of the hangup option:*

> *TTY port opened: /dev/ttyACM0*
> *TTY port speed: 115200*
> *TTY lock file: /var/lock/lockdev/LCK..ttyACM0*
> *TTY port control signals enabled*
> *Caller ID from a modem and optional gateways*
> *Handles modem calls without Caller ID*
> *Modem initialized.*
> *Modem Identifier: CX93001-EIS_V0.2013-V92*
> *Modem country code: B5 United States*
> *Modem ACTIVE PROFILE:*
> *B1 E1 L2 M1 N0 Q0 T V1 W0 X4 Y0 &C1 &D2 &G0 &J0 &K3 &Q5 &R1 &S0 &T5 &X0*
> *S00:0 S01:0 S02:43 S03:13 S04:10 S05:8 S06:2 S07:50 S08:1 S09:6*
> *S10:14 S11:85 S12:50 S18:0 S25:5 S26:1 S36:7 S38:20 S46:138 S48:7*
> *S95:0*
> *Modem supports Data Mode*
> *Modem supports FAX Mode 1*
> *Modem supports FAX Mode 2*
> *Modem supports VOICE Mode*
> *Hangup option = 3 - play an announcement then hangup on a blacklisted call*
> *Internal Hangup recording file: /usr/share/ncid/recordings/DisconnectedNotInService.rmd*
> *Manufacturer: CONEXANT*
> *Modem Voice Sampling Methods:*
> *0,"SIGNED PCM",8,0,8000,0,0*
> *1,"UNSIGNED PCM",8,0,8000,0,0*
> *129,"IMA ADPCM",4,0,8000,0,0*
> *130,"UNSIGNED PCM",8,0,8000,0,0*
> *131,"Mu-Law",8,0,8000,0,0*
> *132,"A-Law",8,0,8000,0,0*
> *133,"14 bit PCM",14,0,8000,0,0*
> *Modem Voice Sampling Method selected: AT+VSM=130*

Hangup Appendix B: Creating a Voice File from Scratch

> *You may wish to check the Network Caller ID page at Wikipedia to see if someone has already documented the steps for your modem.*

> ***Prerequisites***

> *This procedure uses Linux to convert the files.*

> *The **sox** and **mgetty-voice** packages need to be installed.*

> *An audio file, preferably one channel (mono), a sample rate of 8000 Hz (8 kHz) and a sample size of 8 bits. This file can be either of the following:*

> - *An audio file in a format that **sox** supports. A .wav file is the most common. **Sox** does not support .mp3 files.*
> - *A Portable Voice Format (.pvf) file. A few are supplied with NCID.*

> *The chipset identifier returned by the modem's **ATI3** response.*

> - *This will be in the startup section of /var/log/ncidd.log with the line, "Modem Identifier: ...".*

> *The **Voice Sampling Methods** returned by the modem's **AT+VSM=?** response and the directory to store the Raw Modem Data (.rmd) file.*

> - *These are both in the startup section of /var/log/ncidd.log where the lines start with, "Modem Voice Sampling Methods: ..." and "Internal Hangup voice file: ...".*

> - *If you don't see these in the log file, you probably don't have Announce Hangup configured yet in ncidd.conf. You can temporarily enable it by typing the following commands and then examine ncidd.log:*

```
sudo pkill ncidd
sudo ncidd -Dv1 -p 3334 -N1 -H3
```

> ***Examine the Modem's Voice Sampling Methods (VSM)***

> *Your ultimate goal in creating a custom voice file is to take a Portable Voice File (.pvf) and use the **pvftormd** command line program to convert it to a Raw Modem Data (.rmd) file that is specific to your modem type (chipset).*

> *The tricky part is in trying find a compression method for **pvftormd** that is comparable to one of the **Voice Sampling Methods** and codecs returned by the modem's (often cryptic) **AT+VSM=?** response. Compounding the challenge are the facts that:*
>
> - *Different modem manufacturers often have their own codec naming convention when listing the VSMs.*
>
> - *Determining which **pvftormd** modem type is a match for your modem is sometimes not obvious.*
>
> *In the discussion below, we'll see examples of this challenge as we use one of the .pvf files supplied with NCID to create the .rmd file that works on two different modem chipsets: Conexant CX93001 and USR5637.*

> ***Conexant CX93001***

> *The **pvftormd** tool requires a modem type. The supported modem types and raw modem data formats are listed with the **-L** option:*

```
pvftormd -L
```

> *Refer to Hangup Appendix C: Raw Modem Data Formats. You'll notice that there's nothing listed in Column 1 that has "Conexant" as part of its name.*
>
> *So what modem type was used to create the .rmd files supplied with NCID and why/how was it chosen? The answer to the first part of the question is "V253modem" and the answer to the second part is "by experimenting" (or perhaps, "luck").*

> *If your modem isn't listed, which is highly likely, you have to start somewhere so try something for an 8 bit PCM format:*

> - ***V253modem 8** for linear unsigned PCM, or*
> - ***V253modem 9** for linear signed PCM.*

> *Next, you want to look at the modem's VSM info to see if you can find a match on 8 bit, linear, PCM, signed or unsigned.*

> *Here's the VSM info from ncidd.log for a modem with a Conexant CX93001 chipset:*

```
0,"SIGNED PCM",8,0,8000,0,0
1,"UNSIGNED PCM",8,0,8000,0,0
129,"IMA ADPCM",4,0,8000,0,0
130,"UNSIGNED PCM",8,0,8000,0,0
131,"Mu-Law",8,0,8000,0,0
132,"A-Law",8,0,8000,0,0
133,"14 bit PCM",14,0,8000,0,0
```

> *The line for 130 looks promising so we'll try that first. The line for 1 might also work since it "looks" the same as line 130. Probably the only reason 130 was chosen is that users seem to have better luck with values 100 and greater.*

> *In conclusion, we've decided to try these settings for the Conexant CX93001 chipset:*

- *"V253modem 8" for the **pvftormd** tool to create the .rmd*
- *"AT+VSM=130" as the **audiofmt** setting in ncidd.conf*

> ***USR5637***

> *For the USR5637 chipset, it just so happens that trying the .rmd for the Conexant CX93001 chipset works just fine, but it does require a different VSM setting. Looking at ncidd.log for this modem we see:*

```
128,"8-BIT LINEAR",(7200,8000,11025)
129,"16-BIT LINEAR",(7200,8000,11025)
130,"8-BIT ALAW",(8000)
131,"8-BIT ULAW",(8000)
132,"IMA ADPCM",(7200,8000,11025)
```

> *The Conexant CX93001 codec we picked to try was "UNSIGNED PCM" but as you can see, that's not an option for the USR5637. We'll pick line 128, "8-BIT LINEAR" and hope it'll work.*

> *A little more work needs to be done, though. The presence of the () for line 128 means we have to specify one of the three sampling rates, but which one? Recall the line we used for the Conexant CX93001:*

```
130,"UNSIGNED PCM",8,0,8000,0,0
```

> *The sampling rate is fixed at 8000 so we'll try that for the USR5637.*

> *In conclusion, we've decided to try these settings for the USR5637 chipset:*

- *"V253modem 8" for the **pvftormd** tool to create the .rmd*
- *"AT+VSM=128,8000" as the **audiofmt** setting in ncidd.conf*

> ***Creating a Portable Voice File (.pvf)***

> *You would do the steps here if you want to use a .wav file or other audio file format supported by **sox**.*

> *You can either record an announcement or download a .wav file from the internet. A good place to start is [This Is a Recording](#).*

> *Once you have the .wav file, you need to convert it to a .pvf. We'll use the parameters listed under Prerequisites: one channel (mono), a sample rate of 8000 Hz (8 kHz) and a sample size of 8 bits.*

> *Assuming the file is called **custom.wav**:*

```
sox custom.wav -t pvf -c 1 -r 8000 -b 8 custom.pvf
```

> *You can use the **play** command to listen to it:*

```
play custom.pvf
```

> *Make sure the playback is clear and no spoken words get dropped. If the playback doesn't sound good as a .pvf, it is probably not going to sound good when you convert it to an .rmd for playback through the modem. You may need to use a different source .wav file and/or you'll need to experiment with the **sox** parameters.*

> > *Note: It is probably not going to work very well if you try the the **play** command on a virtual Linux machine because playing back audio can cause a performance hit on the virtual machine's CPU and other resources. Instead, you will want to **play** it on a physical machine. As an alternative to the **play** command, you could play it back using cross-platform versions of [Audacity](#) or [VideoLAN](#).*

> *Generally speaking, you only need to create a .pvf file once. It can then be used to create .rmd files for multiple modem chipets.*

> *If you need to examine the properties of a .pvf file, use the **soxi** tool:*

```
$ soxi CallingDeposit.pvf
Input File : 'CallingDeposit.pvf'
Channels : 1
Sample Rate : 8000
Precision : 8-bit
Duration : 00:00:09.90 = 79203 samples ~ 742.528 CDDA sectors
File Size : 79.2k
Bit Rate : 64.0k
Sample Encoding: 8-bit Signed Integer PCM
$
```

> ***Creating a Raw Modem Data (.rmd) File***

> *Once you have a .pvf file, you must convert it to an .rmd that is specific for your modem's chipset.*

> *Assuming the file is called **custom.wav**:*

```
pvftormd V253modem 8 custom.pvf custom.rmd
```

> *The **DisconnectedNotInService.pvf** voice file was used to create **DisconnectedNotInService.rmd**. The .pvf files for the distribution are in the documentation directory which is usually:*

> > */usr/share/doc/ncid/recordings*
>
> > *or*

> > *`/usr/local/share/doc/ncid/recordings`*

> *Use **DisconnectedNotInService.pvf** to create a **raw modem data (.rmd)** file for your modem if the default one, **DisconnectedNotInService.rmd**, is not usable.*

> *Convert it to .rmd:*

```
pvftormd V253modem 8 DisconnectedNotInService.pvf
DisconnectedNotInService.rmd
```

> *If you need to examine the properties of an .rmd file, use the **rmdfile** tool:*

```
$ rmdfile CallingDeposit.rmd
CallingDeposit.rmd: RMD1
modem type is: "V253modem"
compression method: 0x0008
sample speed: 8000
bits per sample: 8
$
```

> ***Configuring NCID to use a Custom Voice File***

> *Copy the custom .rmd file to the directory to store the Raw Modem Data (.rmd) file (see Prerequisites).*

> *Edit **ncidd.conf** to enable Announce Hangup, indicate the name of the custom .rmd file and the AT+VSM command to use.*

> *Example:*

```
set hangup = 3
set announce = custom.rmd
set audiofmt = "AT+VSM=128,8000"
```

> ***Testing a Custom Voice File***

> *The best way to test a custom voice file is to temporarily add your phone number to the blacklist file and call yourself. Use a handset or headset to listen to the call and not speakerphone or handsfree mode.*

> *If you experience any of the following, you will need to experiment with different parameters when running **pvftormd** and/or different AT+VSM settings:*

> - *dropped spoken words or words that are completely unrecognizable*
> - *play back is too fast or too slow*
> - *you hear static noise*

Hangup Appendix C: Raw Modem Data Formats

> *You would probably refer to this appendix only if you're using the Announce Hangup option.*

> *The **pvftormd** command line program requires a modem type. The supported raw modem data formats are listed with the **-L** option:*

> ***pvftormd -L***

> *Column headings added to chart for readability:*
>
> 1. *Modem type or manufacturer.*
> 2. *One or more numbers representing different compression methods (these values are unique to **pvftormd** and have no relation to a modem's compression number in its **AT+VSM=?** response).*
> 3. *Description of the compression method and usually lists the bit levels supported by **pvftormd**.*

> *Output of **pvftormd -L** follows:*

```
pvftormd experimental test release 0.9.32 / with duplex patch
```

```
supported raw modem data formats:
```

Column 1	***Column 2***	***Column 3***
Digi	*4*	*G.711u PCM*
Digi	*5*	*G.711A PCM*
Elsa	*2, 3, 4*	*2/3/4-bit Rockwell ADPCM*
ISDN4Linux	*2, 3, 4*	*2/3/4-bit ZyXEL ADPCM*
ISDN4Linux	*5*	*G.711A PCM*
ISDN4Linux	*6*	*G.711u PCM*
Lucent	*1*	*8 bit linear PCM*
Lucent	*2*	*16 bit linear PCM*
Lucent	*3*	*G.711A PCM*
Lucent	*4*	*G.711u PCM*
Lucent	*5*	*4 bit IMA ADPCM*
MT_2834	*4*	*4 bit IMA ADPCM*
MT_5634	*4*	*bit IMA ADPCM*
Rockwell	*2, 3, 4*	*2/3/4-bit Rockwell ADPCM*
Rockwell	*8*	*8-bit Rockwell PCM*
UMC	*4*	*G.721 ADPCM*
US_Robotics	*1*	*USR-GSM*
US_Robotics	*4*	*G.721 ADPCM*
V253modem	*2, 4*	*2/4-bit Rockwell ADPCM*
V253modem	*5*	*4-bit IMA ADPCM*
V253modem	*6*	*G.711u PCM*
V253modem	*7*	*G.711A PCM*

V253modem	*8*	*8-bit linear unsigned PCM*
V253modem	*9*	*8-bit linear signed PCM*
V253modem	*12*	*16-bit linear signed PCM Intel Order*
V253modem	*13*	*16-bit linear unsigned PCM Intel Order*
ZyXEL_1496	*2, 3, 4*	*2/3/4-bit ZyXEL ADPCM*
ZyXEL_2864	*2, 3, 4*	*2/3/4-bit ZyXEL ADPCM*
ZyXEL_2864	*81*	*8-bit Rockwell PCM*
ZyXEL_Omni56K	*4*	*4-bit Digispeech ADPCM (?)*

example:

pvftormd Rockwell 4 infile.pvf outfile.rmd

Hangup Appendix C: Modem AT+VSM Command Explained

Here is a breakdown of what each parameter means in the AT+VSM command:

```
Command: AT+VSM=?
Response: <cml>,<cmid>,<bps>,<tm>,<vsr>,<sds>,<sel>
```

```
<cml> Decimal number identifying the compression method (1, 129,
      130, 140, or 141).
```

```
<cmid> Alphanumeric string describing the compression method (UNSIGNED
       PCM, IMA ADPCM, UNSIGNED PCM, 2 Bit ADPCM, or 4 Bit ADPCM).
```

```
<bps> Decimal number defining the average number of bits in the
      compressed sample not including silence compression (2, 4 or 8).
```

```
<tm> Decimal number (0) reporting the time interval, in units of
     0.1 second, between timing marks. A value of 0 reports that
     timing marks are not supported.
```

```
<vsr> <range of values> containing the supported range of voice
      samples per second of the analog signal (8000).
```

```
<scs> <range of values> containing the supported range of sensitivity
      settings for voice receives (0). A 0 indicates not supported.
```

```
<sel> <range of values> containing the supported range of expansion
      values for voice transmits (0). A 0 indicates not supported.
```

Server Extensions

> *Table of Contents*

Description

> *The ncidd server supports **Server Extensions**. A Server Extension is an external script or program that is called by ncidd to perform a function and return a result. Server Extensions are a way for users to add functionality to NCID without requiring changes to NCID itself. Server Extensions are isolated from the main NCID distribution and because of this they do not normally require any changes when NCID is upgraded to a later version.*

> *You can use any scripting or programming language desired.*

Hangup Extension

> *The first Server Extension distributed with NCID is the **Hangup Extension**.*

> *A Hangup Extension can be used with and without Internal Hangup. (Internal Hangup is defined as call termination using the `ncidd.blacklist` and `ncidd.whitelist` files.)*

> *It works like this:*

> > *When Caller ID is received from a modem and it is not terminated by the Internal Hangup logic and is not in 'ncidd.whitelist', the server will check to see if a Hangup Extension has been enabled. If so, the server will pass the current call info to the Hangup Extension, execute it and wait for a response. If the Hangup Extension results in a positive match by returning the word **hangup**, call termination will take place automatically.*

> > *The Hangup Extension can also change the "announce" file name to give a different message to some callers, and can return a hangup reason that will be added to the displayed caller name.*

> *Hangup Extensions use all of the same Internal Hangup modes (normal, fax, announce). Be sure to review the required Modem Configuration parameters.*

> *Three settings in **ncidd.conf** control Hangup Extensions:*

Setting	*Values*
set hupmode	*0 = disabled* *1 = Normal Hangup* *2 = Fax Hangup* *3 = Announce Hangup* *default: 0*
set hupname	*the name of the hangup script or program*

	default: hangup-nohangup
set huprmd	*optional voice file to be played if hupmode = 3* *default: "announce" file used by Internal Hangup (usually **DisconnectedNotInService.rmd**)*

*When you enable a Hangup Extension, you also need to indicate the hangup script to use. The default is **hangup-nohangup** which will run, but it will never trigger a hangup. When testing your script you can give the full path name to where it is located. When it's working OK, copy it to the path appropriate for your operating system:*

/usr/share/ncid/extensions

or

/usr/local/share/ncid/extensions

Once the file is there you give the name of the script in ncidd.conf:

set hupname = hangup-custom

You create your own Hangup Extension script/program external to NCID in whatever language you would like. It can have whatever logic you wish for terminating a call. You can even have it return the name of a customized voice message file for individual phone numbers. It is not necessary for your Hangup Extension to check `ncidd.whitelist` because the NCID server already does this for you and will have automatically applied any Simple Expressions or Regular Expressions to the incoming call.

The technical details of creating a Hangup Extension are outside the scope of this document (see NCID-API).

However, several ready-to-use Hangup Extensions and template Hangup Extensions are included with NCID. Template Hangup Extensions have a file name ending in '-skel' (short for "skeleton"). These will always be replaced when NCID is updated. Before customizing a template, it is essential that you copy or rename the template script so that it does not end in '-skel'.

Below is a summary of the Hangup Extensions included with NCID:

- *The hangup-calls extension hangs up on every call not in the whitelist.*
- *The hangup-closed-skel script is a template for playing a recorded "we are closed" message prior to hanging up on calls within a specific time period. You need to customize the start and end times and record a message. It is recommended that your recorded message include the name of your business so callers can be sure they dialed the correct number.*
- *The hangup-combo-skel script is a template for calling two extension scripts. By default, it calls hangup-fakenum and hangup-fcc. Rename it to hangup-combo after you have customized it.*
- *The hangup-fakenum extension hangs up on calls where the number is absent or not allowed in the North American numbering plan, or where the name is mostly numeric. Optionally, it will also check a list of valid area codes brought in with the get-areacodes-list script. Read more at Enforcing the North American Numbering Plan.*

- *The hangup-fcc extension hangs up on calls where the number can be found in the FCC's open data list of "Unwanted Calls". The blacklist is stored locally in /etc/ncid/fcc.blacklist and is typically updated daily by a cron job (or Mac OS X launchd daemon) that calls get-fcc-list to fetch the latest version from a web site. Read more at CallerID Blacklist from FCC Data.*

- *The hangup-message-skel script is a template that contains a sample list of phone numbers with their corresponding name of a recorded message (voice) file that is played before terminating the call. There are no provisions to record a message from the caller; NCID is not an answering machine.*

- *The hangup-nohangup script does not hangup on any calls. This is the default script in case the user forgets to change the hupmode setting in **ncidd.conf**.*

- *The hangup-postal-code extension scrip hangs up of calls if a search of the US postal codes for a state, commonwealth or territory abbreviation at the end of a Caller ID name if found.*

- *The hangup-skel extension is a very basic template to determine if the number or name received should tell ncidd to hangup. You might use this as a starting point, for example, to query an Internet site to determine if the call should be terminated. Or perhaps you have a local relational database with names or numbers of blacklisted callers.*

Log Files

Requirement

- ***logrotate***

*NCID uses **logrotate** to prune its log files each month. When it prunes the log, it saves a backup. Up to five backups are saved.*

*Another system log rotation program can be used but it must be configured to zero (empty) the log each month in order to use the **ncid-yearlog** program.*

Description

Here is an alphabetical list of NCID log files stored in /var/log:

Log file name	*Log type*	*Description*
cidcall.log	*server*	*Calls and messages*
ciddata.log	*server*	*Raw data received by ncidd server as captured from modems and gateways**
lcdncid.log	*client*	*LCDproc client activity*
ncid2ncid.log	*gateway*	*NCID-to-NCID gateway activity*

ncidd.log	*server*	*Server activity*
obi2ncid.log	*gateway*	*OBihai gateway activity*
rn2ncid.log	*gateway*	*Remote Notifier gateway activity*
sip2ncid.log	*gateway*	*SIP gateway activity*
wc2ncid.log	*gateway*	*Whozz Calling gateway activity*
xdmf2ncid.log	*gateway*	*XDMF gateway activity*
yac2ncid.log	*gateway*	*YAC gateway activity*

**Note: If /var/log/ciddata.log exists, ncidd will write to it. It must be manually created prior to launching ncidd:*

```
sudo touch /var/log/ciddata.log
```

*Each month **logrotate** uses /etc/logrotate.d/ncid to prune files as required in /etc/ncid/ncidrotate.conf. Only two variables are expected to change: RotateOff and Lines2keep.*

If the user does not want a log rotated, set RotateOff=1. This will let the log keep growing until the operating system decides it is too large.

The default for Lines2keep is 0. Some users like to keep some lines in the log when it is pruned. If you would like to keep the last 10 lines at the start of the month, set Lines2keep=10

If you turned rotation off or do not prune a log to zero each month, you should backup the log to someplace that is not /var.

Yearly Logs

NCID can keep yearly logs automatically in $HOME/ncid/log/ by running /usr/share/ncid/sys/ncid-yearlog on the first of every month from the user's crontab.

*If **ncidrotate.conf** has Lines2keep=0 and **rotatebysize.conf** has minsize 1, you can enable **ncid-yearlog** by creating or editing a crontab and adding this line (NOTE: /usr/share can be /usr/local/share on some operating systems):*

```
11 5 1 * * /usr/share/ncid/sys/ncid-yearlog
```

Command Line Tools

Table of Contents

Tools Index

get-areacodes-list
ncid-yearlog
ncidutil
update-cidcall

Overview

NCID has command line Perl scripts (also called tools) that can list or modify the ncidd.alias, ncidd.blacklist, ncidd.whitelist and cidcall.log files.

There are four tools for dealing with the alias, blacklist and whitelist files:

cidalias, cidcall, cidupdate and ncidutil.

If you edit and modify ncidd.alias, ncidd.blacklist, or ncidd.whitelist with an editor:

- *(optional) Run cidupdate after modifying ncidd.alias to update the cidcall.log file with the new aliases*
- *Force the server to reload ncidd.alias, ncidd.blacklist and ncidd.whitelist:*

sudo pkill --signal SIGHUP ncidd

NCID has a tool for creating a yearly call log from the monthly call logs. The yearly call log is updated each month. The monthly call logs are only kept for a specific period of time but yearly call logs are kept until deleted.

cidalias

*The **cidalias** tool displays aliases in the alias file in one of three different formats: raw, human readable and delimited.*

See the cidalias.1 man page for a complete description and all options.

cidcall

*The **cidcall** tool is used to view either the cidcall.log file or the cidcall-.log in one of two different formats: raw and human readable. The default is to display BLK, CID, HUP, OUT, PID and WID lines from the cidcall.log file in a human readable format. Messages and Smartphone Notes will be viewed when their option is selected.*

See the cidcall.1 man page for a complete description and all options.

EXAMPLES:

View all call types, but not message types:

```
cidcall
```

View messages and notes:

```
cidcall --MSG --NOT
```

View all call types from the 2018 call log:

```
cidcall --yearlog 2018
```

> *Assuming the current year is 2019, view all call types from up to but not including the current month in the cidcall.log:*

```
cidcall --year 2019
```

cidnumber-info

> *The **cidnumber-info** tool looks up a phonenumber and shows the number formatted for the country of the number along with the following name and data fields in the ncidd call log format*

cidupdate

> *The **cidupdate** tool is used to update the cidcall.log file with newly created aliases. It is also used by the server whenever clients want the call logfile updated.*

> *Command Line Usage:*

- *Add one of more aliases to ncidd.alias*
- *Run **cidupdate** to update cidcall.log for any calls that require the new alias or aliases.*
- *Reload ncidd.alias, ncidd.blacklist and ncidd.whitelist:*

> > *sudo pkill --signal SIGHUP ncidd*

> *See the cidupdate.1 man page for a complete description and all options.*

get-fcc-list

> *The **get-fcc-list** script is usually called daily from a crontab to fetch the `fcc.blacklist` from a server for use by hangup-fcc or to be added to the ncidd.blacklist.*

> *It is more completely described in its own section about using the CallerID Blacklist from FCC Data*

get-areacodes-list

> *The **get-areacodes-list** script only needs to be called every six months or less to fetch a newer list of valid area codes for hangup-fakenum. It can be added the a crontab if desired.*

> *It is more completely described in its own section about using the CallerID Blacklist from FCC Data*

ncid-yearlog

> *The **ncid-yearlog** tool automatically creates a yearly call log from the monthly call logs. It is called from the user's crontab on the first of the month. Review Yearly Logs and ncid-yearlog.1*

ncidutil

> *The **ncidutil** tool is only used by the server to add, modify, or remove entries from ncid.alias. It is also used by the server to add or remove entries from ncidd.blacklist and ncidd.whitelist.*

> *See the ncidutil.1 man page for a complete description and all options.*

update-cidcall

> *The **update-cidcall** tool adds the new name and data fields from **ncidnumber-info** to the lines from the **cidcall.log** file. The output is sent to STDOUT.*

CallerID Blacklist from FCC Data

> *Table of Contents*

Description

> *The FCC gets many complaints about unwanted callers and makes a summary available to the public. We refer to this summary as the FCC Data.*

> *When used with a modem (see the Hangup Overview), **ncidd** can use this information as a blacklist to hang up on unwanted North American callers.*

> *A **get-fcc-list** script, run periodically, fetches this data from an intermediate web server.*

> *The FCC Data can be used with the NCID server's Internal Hangup and Hangup Extensions.*

Requirements

> - *NCID release 1.7 or newer*
> - *a modem to do the hangup*
> - *internet connection*
> - ***wget** - fetches files using http*
> - ***pkill** - send signal to all running ncidd servers to reload alias, blacklist and whitelist*

Background

> *On Oct 21, 2015, the United States Federal Communications Commission (FCC) announced (archived here or here) they would be releasing weekly data to support robocall blocking technologies.*

> *On May 23, 2016, the FCC announced (archived here or here) a new method for us to obtain the information. It appears to be updated in either real time or daily.*

> *You may also [go to the FCC's site and add your own complaints.]*

> *The FCC Data is a huge spreadsheet available from the FCC's Open Data project. It has a row for each complaint received about annoying phone callers going back to October 2014.*

> *The spreadsheet is growing larger as complaints arrive at a rate of around 500 per day. There is a lot of information that we don't care about and some complaint types that don't involve phone calls.*

> *You can find the CGB - Consumer Complaints Data at [https://opendata.fcc.gov/Consumer/CGB-Consumer-Complaints-Data/3xyp-aqkj]*

> *The original NCID script to fetch and analyze the complaints data was written by Mike Stember. It was named **FCC2ncid**.*

> *A change in the FCC's data caused **FCC2ncid** to stop finding new numbers after October 2016, so it has been replaced by **get-fcc-list**.*

How FCC Data Retrieval Works

> *Every day at around 7:11AM EST, an intermediate web server downloads the FCC's large dataset. By performing the following processing, the intermediate web server reduces a 22 megabyte download to a more manageable 195,000 bytes:*

> - *duplicates are merged together and counted*
> - *only numbers reported three or more times are included*
> - *numbers which have been inactive for 600 days are excluded*
> - *the resulting list is sorted and stored as `fcc.blacklist`*

> *Each installation of NCID can run a local copy of the **get-fcc-list** script to fetch `fcc.blacklist` .*

> *If the script is configured for use with NCID Hangup Extensions (usually **hangup-fcc**), no further processing is needed.*

> *If the script is configured for use with the NCID server's Internal Hangup, it performs the following additional steps:*

> - *backs up the existing `ncidd.blacklist` into `/var/backups/ncid/`*
> - *deletes the old FCC data from `ncidd.blacklist`*
> - *appends the latest FCC data to `ncidd.blacklist`*
> - *signals **ncidd** to reread the alias, blacklist and whitelist files*

Configure NCID

> *Choose one of the following:*

> - *Internal Hangup*

> *File `fcc.blacklist` is appended to `ncidd.blacklist` . This method allows NCID clients (e.g., NCIDpop, NCID Android) to query the blacklist.*

> 1. *Configure the NCID server to use [Internal Hangup] as explained in the **[User Manual]**.*
> 2. *Configure **get-fcc-list** for periodic update with the -a (append) option.*

> - *Hangup Extension*

> *File `fcc.blacklist` is used as-is by the Hangup Extension. This method has a smaller memory footprint and is more modular.*
>
> 1. *Configure the NCID server to use Hangup Extensions as explained in the **User Manual**. Specify **hangup-fcc** as the name of the extension.*
> 2. *Configure **get-fcc-list** for periodic update with the -n (no append) option.*

> *If you wish to transition from using the FCC Data with Internal Hangup to using Hangup Extensions:*
>
> - *Run **get-fcc-list** as root, just once, from the command line, using the -r option.*
>
> *This will backup the current `ncidd.blacklist`, remove old FCC Data lines and force the server to reread the modified `ncidd.blacklist`.*
>
> - *You must specify the full path to the script.*
>
> ```
> sudo /usr/share/ncid/sys/get-fcc-list -r
> ```
>
> *or*
>
> ```
> sudo /usr/local/share/ncid/sys/get-fcc-list -r
> ```

Configure get-fcc-list for Periodic Update

> *You likely want to run **get-fcc-list** as a daily background task. It is suggested that it be run at 08:15AM local time.*

> - *Linux/UNIX - use **cron** and log activity to `/tmp/get-fcc-list.log`*
>
> 1. *Run the **crontab** editor as root:*
>
> ```
> sudo crontab -e
> ```
>
> 2. *Add two new lines at the end depending on the hangup method:*
>
> - *Internal Hangup - append (-a) FCC Data to `ncidd.blacklist`*
>
> ```
> # run 1x each day at 08:15
>
> 15 08 * * *   /usr/share/ncid/sys/get-fcc-list -a \
>               > /tmp/get-fcc-list.log 2>&1
> ```
>
> - *Hangup Extension - do not append (-n), use `fcc.blacklist` as-is*
>
> ```
> # run 1x each day at 08:15
>
> 15 08 * * *   /usr/share/ncid/sys/get-fcc-list -n \
>               > /tmp/get-fcc-list.log 2>&1
> ```

Enforcing the North American Numbering Plan

Table of Contents

Description
Requirements
Background
Configure NCID
Configure get-areacodes-list for Periodic Update

Description

The North American Numbering Plan has some rules which many spam callers have not figured out, or can not duplicate.

*When used with a modem (see the Hangup Overview), **ncidd** can use this to hang up on unwanted callers.*

*optionally, a **get-areacodes-list** script, run periodically, updates a list of valid area codes.*

Requirements

- *NCID release 1.7 or newer*
- *a modem to do the hangup*
- *ten digit caller ID*

Creating and updating the list of valid area codes requires:

- *internet connection*
- ***wget** - fetches files using http*

Background

Many calling phone numbers do not exist in North America. Spam callers from distant places don't seem to know this and call with area codes like 000 or exchange codes like 111.

*In addition, spam callers appear to have difficulty filling in the caller ID name field. This will show up as something like V1234567890 or be the same as the calling number. Phone companies supply text like A SMITH for the name, so **hangup-fakenum** disallows numeric names.*

*The **hangup-fakenum** extension runs after the built-in aliases, and the configured aliases in **ncidd.alias** have been applied. The whitelist is checked, then (if hangup is enabled) the blacklist, then (if hupmode is enabled) the extension script.*

Configure NCID

Hangup Extension

1. *Configure the NCID server to use Hangup Extensions as explained in the **User Manual**. Specify **hangup-fakenum** as the name of the extension (or chain some extensions together with hangup-combo).*
2. *Add a list of valid area codes by running **get-areacodes-list** once and then schedule it for periodic update.*

```
sudo /usr/share/ncid/sys/get-areacodes-list
```

or

```
sudo /usr/local/share/ncid/sys/get-areacodes-list
```

Configure get-areacodes-list for Periodic Update

*You might want to run **get-areacodes-list** as a repeated background task. The area codes do not change very often.*

- *Linux/UNIX - use **cron** and log activity to `/tmp/get-areacodes-list.log`*

1. *Run the **crontab** editor as root:*

```
sudo crontab -e
```

2. *Add two new lines at the end:*

```
# run 2x yearly at 08:15

15 08 3 6,12 *   /usr/share/ncid/sys/get-areacodes-list \
             > /tmp/get-areacodes-list.log 2>&1
```

NCID FAQ

Table of Contents

FAQ Index

General

- **What is Caller ID?**

This is best explained in this Caller ID article on Wikipedia.

- **What is NCID?**

NCID is a Network Caller ID package that distributes Caller ID over a network to a variety of devices and computers.

NCID consists of:

- *A server (`ncidd`) that normally uses a device to monitor a telephone line for CID information.*
- *A Universal Client (`ncid`) that obtains the CID information from the NCID server and displays it.*
- *Multiple gateways that obtain the Caller ID information and send it to the NCID server as a CID message.*
- *Command line tools that deal with the Whozz Calling (WC) Ethernet Link device and edit or list the **cidcall.log**, **ncidd.alias**, **ncidd.blacklist** and **ncidd.whitelist** files.*

The Universal Client has output modules that can be used to push CID to other computers and devices. Here is a partial list:

- *MythTV*
- *pager*
- *cell phone, using the email-to-SMS gateway of the carrier*
- *smartphones and tablets running Android or iOS*
- *"speak", using a text-to-speech converter*
- *LCD displays*

- **Does NCID only support one client?**

The NCID package only comes with one client, but other clients are available such as `NCIDPop` and `LCDncid` .

Third party clients are also available for various platforms:

- *OSX Pop-Up*
- *NCIDStatusBarMenu*
- *Yet Another Caller ID Program*

> - *TWNcid*
> - *NCID client for GNOME*
> - *ncid-sqlite*
> - *googletv_ncid*
> - *SmartCallMonitor*
> - *CIDTracker*
> - *ncidApp*
> - *tellows Callblocker*
> - *Sipura2NCID*
> - *NCID Sender*
> - *ncid-xmpp*
> - *ncid-client-py*
> - *ncid-notify*
> - *liveNCID*
> - *ncid-psql*
> - *ncidhitta*
> - *ncidmon*
> - *node-ncid-client*
> - *etc.*

- **Can I have multiple servers and clients?**

> *There should only be one server per phone line.*

> *NCID is designed to have multiple clients. For example: a client on a cell phone, a client on each computer in your network and a client to handle your cell phone.*

- **Can I run the NCID client under Windows?**

> *The GUI client is the only part of NCID that runs directly under Windows. Output modules are not supported.*

> *If you want to run the complete NCID distribution, you need to install Cygwin. You need to configure the NCID server so it will only use its gateways to obtain the Caller ID. If you want to use a modem, you need to also install a YAC server.*

- **How do I determine if my modem supports Caller ID?**

> *The modem documentation should tell you if the modem supports Caller ID and how to set it up. NCID will try two ways to configure a modem for Caller ID and the configuration file (**ncidd.conf**) has other methods you can try.*

> *NCID documentation has a section called Modems that gives some information on configuring a modem for Caller ID and there is also a section called Modem Caller ID Test that tells how to use ncidd to test the modem.*

- **What is an NCID gateway?**

> *An NCID gateway obtains the Caller ID information and sends it to the NCID server as a CID message. Currently included gateways are:*

> - *An XDMF Gateway that obtains the CID information as messages from an SDMF or MDMF USB device, such as the CTI Comet USB or the Holtek HT9032D based PSTN Caller ID module and sends it to the NCID server. Some modems can also be configured to output in SDMF/MDMF format.*
> - *A SIP Gateway that obtains the CID information using VoIP SIP packets and sends it to the NCID server.*
> - *A YAC (Yet Another Caller ID) Gateway that obtains the CID information from a YAC modem server.*

> - *A Whozz Calling (WC) Ethernet Link device gateway that obtains the CID information from multiple POTS (Plain Old Telephone Service) lines.*
> - *A gateway for the Android app "Remote Notifier" that obtains smart phone CID and text messages.*
> - *An NCID-to-NCID Gateway that sends the CID information from one or more NCID servers to a selected NCID server.*

- **Can NCID be used with YAC (Yet Another Caller ID)?**

> *Yes. NCID has a YAC output module and a YAC Gateway.*

> *The YAC output module is used with the NCID client to obtain the CID information from the NCID server and send it to YAC listeners.*

> *The YAC gateway receives the CID information from a YAC modem server, formats it and sends it to the NCID server as a CID message. If your windows PC has a modem, you can install YAC and configure it to send the CID to the YAC Gateway.*

- **Does NCID support more than one phone line?**

> *Yes, NCID supports any/all of the following:*

> - *Up to 5 modems or serial devices*
> - *Multiple SIP Gateways*
> - *Multiple YAC Gateways*
> - *Multiple Whozz Calling (WC) devices using the WC gateway*
> - *Multiple XDMF Gateways on different, network-connected remote machines, or on the same, local machine, provided a separate configuration file or the --usbport|-u command line option are used for each gateway.*
>
>   *It should be noted that multiple XDMF Gateway setups are not tested due to lack of hardware units, therefore neither officially supported.*

> *Each SIP Gateway can support multiple VoIP telephone connections.*

> *Each WC device can support either 2, 4, or 8 POTS (Plain Old Telephone Service) lines.*

> *A single XDMF Gateway supports only one POTS line, however, you can have multiple XDMF Gateways each connected to a separate POTS line.*

- **What packages are distributed?**

> - *NCID: Package includes a server, client and gateways.*
> - *LCDncid: An NCID client that outputs to a LCD display using LCDproc.*
> - *NCIDPop: An NCID popup client for Linux based distros, Mac OS X and Windows.*

Server

- **What is an "alias" and how do I configure one?**

> *An "alias" allows you to replace a generic Caller ID name with a custom one that is more meaningful. You can configure several hundred aliases if you want. Aliases are stored in the **ncidd.alias** file.*

> *For example, if an incoming call has the name "WIRELESS CALLER" you can use an alias to change it to the real name of the caller. You would use this form of alias:*

```
    alias NAME "FROM" = "TO" if "TELEPHONE_NUMBER"
```

> *Since we do not care what name is there, we will use '*' in the FROM field. The TO field can contain spaces so in our case we want it to say: "John on cell". The most important field is the TELEPHONE_NUMBER; this must match the number ncidd receives and in most cases, it includes a '1', even though it is not displayed. Putting in the values, our alias looks like this:*

```
alias NAME * = "John on cell" if 14075551212
```

> *The complete documentation for aliases is in the **ncidd.alias** man page and as comments in the **ncidd.alias** file.*

- **What is a "blacklisted" caller and how do I configure one?**

> *The blacklist is a list of names or numbers that NCID will automatically hangup. The blacklisted names and numbers are stored in the **ncidd.blacklist** file.*

> *The blacklist supports simple expressions (the default) or extended Posix Regular Expressions (an option).*

> ***Using Simple Expressions:***

> > *The name or number in the blacklist is treated as a substring of the caller name or number. For example, using a 10 digit US number:*

```
3215551212 - will only match 3215551212
321555     - will match any number with 321555 in it
^321555    - will match any number beginning with 321555
```

> ***Using Posix Extended Regular Expressions:***

```
^3215551212$ - will only match 3215551212
.*321555.*   - will match any number with 321555 in it
^321555      - will match any number beginning with 321555
```

> *The **ncidd.blacklist** file comes preconfigured for PRIVATE names, a spoofing area code and a few expensive area codes.*

> *The complete documentation for the blacklist is in the **ncidd.blacklist** man page and as comments in the **ncidd.blacklist** file.*

- **What is a "whitelisted" caller and how do I configure one?**

> *The whitelist is a list of names or numbers in the **ncidd.whitelist** file. If a call matches a name or number in the **ncidd.blacklist** file, the whitelist file is consulted to see if the call should be allowed.*

> *The whitelist file uses the same expressions as the blacklist file.*

> *As an example, the blacklist file comes preconfigured to blacklist the entire 999 unused area code. If you want to allow a specific number from area code 999, you would add it to **ncidd.whitelist**:*

> ***Using Simple Expressions:** 9995551212*

> ***Using Posix Extended Regular Expressions:** 9995551212 **or** ^9995551212$*

> *You might notice there are 2 entries in the blacklist file for each of the blacklisted area codes. This is because in the US some systems send a leading **1** and some do not. Not knowing which system a user might have, two entries are made to work with both types. Thus the above example using simple expressions becomes:*

```
9995551212 19995551212
```

> *The complete documentation for the whitelist is in the **ncidd.whitelist** man page and as comments in the **ncidd.whitelist** file.*

- **How do I configure NCID to auto hangup on specific calls?**

> *The server hangup feature is configured in the "Automatic Call Hangup" section of the **ncidd.conf** file. Just remove the '#' from the line:*

```
    # set hangup = 1
```

> *Once you change **ncidd.conf**, you must start/restart ncidd to read it.*

> *You need to enter the caller name or telephone number in the **ncidd.blacklist** file, usually one name or number per line.*

> *If there is a match on the name or number when a call comes in, it will immediately be terminated. When a call is terminated, an "HUP" entry is made in the **ncidd.log** file.*

Gateways

- **How do I configure NCID to use the XDMF Gateway?**

> *You need an SDMF or MDMF USB device, such as the CTI Comet USB or the Holtek HT9032D based PSTN Caller ID module. Or, if your modem supports it, you can use the XDMF Gateway with a modem configured to output Caller ID in SDMF or MDMF hexadecimal format using ASCII text. The most typical setup uses the CTI Comet USB or the Holtek HT9032D based PSTN Caller ID module and is described below.*

> *You need to configure ncidd and the xdmf2ncid gateway.*

> *Because the POTS (Plain Old Telephone Service) lines are connected directly to the CTI Comet USB device or the Holtek HT9032D based PSTN Caller ID module, you need to set cidinput to 2 or 3 in **ncidd.conf**. Once you change **ncidd.conf**, you must start/restart ncidd to read it.*

> *Then you need to modify **xdmf2ncid.conf** to set the USB port used by the CTI Comet USB or the Holtek HT9032D based PSTN Caller ID module, uncomment the line and change the USB port. For Linux systems, ttyUSB? is normally used where USB? is USB0, USB1, USB2, or USB3.*

```
Edit xdmf2ncid.conf:

change the line: #usbport = /dev/ttyUSB1
             to: usbport = /dev/ttyUSB1
```

> *If using the Holtek HT9032D based PSTN Caller ID module, set **ht9032 = 1**.*

```
Edit xdmf2ncid.conf:

change the line: # ht9032 = 1
             to: ht9032 = 1
```

> *Make sure the CTI Comet USB device or the Holtek HT9032D based PSTN Caller ID module are setup properly. Start xdmf2ncid in test mode:*

```
xdmf2ncid -t
```

- **How do I configure NCID to use the SIP gateway?**

> *You need to configure your network, ncidd and the sip2ncid gateway for the ports that SIP Invite uses.*

> *The sip2ncid gateway automatically monitors both TCP and UDP protocol packets.*

> *You can test for network packets in general, or SIP packets for a particular port, using sip2ncid. You would need to use the -T|--testall or the -t|--testsip option. Once you determine the ports being used, enter them in* ***sip2ncid.conf****.*

> *If you have SIP packets on your home network, your network is already configured and ready to use. However, a home network may need to be configured to receive SIP packets. For a router/phone device you may need to put your computer in the Demilitarized Zone (DMZ) to see SIP packets; usually port forwarding will not work. You should also review this tutorial: [Configuring your home network for SIP-based callerID (on the Wayback Machine)]*

> *If you are using a POTS (Plain Old Telephone Service) line and SIP, no additional ncidd configuration is necessary. If you are only using SIP you need to set cidinput to 2 or 3 in* ***ncidd.conf****. Once you change* ***ncidd.conf****, you must start/restart ncidd to read it.*

- **How do I configure NCID to use the YAC (Yet Another Caller ID) Gateway?**

> *You need to configure ncidd, yac2ncid and your YAC server.*

> *If you are using a POTS (Plain Old Telephone Service) line and YAC, no additional ncidd configuration is necessary. If you are only using YAC, or SIP and YAC, you need to set cidinput to 2 or 3 in* ***ncidd.conf****. Once you change* ***ncidd.conf****, you must start/restart ncidd to read it.*

> *If the YAC server is on the same computer as ncidd, no configuration is necessary. If it is on a different computer, the IP address of NCID needs to be inserted into the* ***yac2ncid.conf*** *file.*

> *The NCID YAC Gateway is a YAC Listener, so the YAC server needs the address IP address of the computer running yac2ncid.*

- **How do I configure NCID to use the Whozz Calling (WC) Gateway?**

> *You need a [Whozz Calling] Ethernet Link device. You can get either a basic or a deluxe 2, 4, or 8 port model. You can configure as many as you would like, in any combination of 2, 4, or 8 port units.*

> *You need to configure ncidd, wc2ncid and the Whozz Calling device.*

> *Because the POTS (Plain Old Telephone Service) lines are connected directly to the Whozz Calling device, you need to set cidinput to 2 or 3 in* ***ncidd.conf****. Once you change* ***ncidd.conf****, you must start/restart ncidd to read it.*

> *If your local network uses 192.168.0.x you do not need to configure* ***wc2ncid.conf****. If you have a different network, say 192.168.1.x, then you need to modify* ***wc2ncid.conf****:*

```
Edit wc2ncid.conf:

change the line: wcaddr = 192.168.0.90
to: wcaddr = 192.168.1.90
```

> *The wc2ncid gateway script needs to be run at least once before using it with NCID. It also needs to be run whenever you change wcaddr in* ***wc2ncid.conf****.*

```
wc2ncid --set-wc
```

> *Make sure the Whozz Calling device is setup properly. Start wc2ncid in test mode:*

```
wc2ncid -t
```

- **How do I configure NCID to use the Remote Notifier Gateway?**

> *You need to install the free [Remote Notifier for Android] app from F-Droid.*

> *You need to configure the Remote Notifier app, ncidd and rn2ncid.*

> *Launch Remote Notifier configure the IP address of the computer running the NCID server.*

> *If you are using a POTS (Plain Old Telephone Service) line and Remote Notifier, no additional ncidd configuration is necessary. If you are only using Remote Notifier, set cidinput to 2 or 3 in **ncidd.conf**. Once you change **ncidd.conf**, you must start/restart ncidd to read it.*

> *Normally rn2ncid does not need to be configured unless you are using ncid-page to send calls and messages to your smart phone. In that case you need to edit the reject line at the end of the **rn2ncid.conf** file: Specify the "from" of SMS/MMS messages to be rejected and not passed through to the NCID server. This is usually an email address dedicated to the SMS-to-email service of your carrier.*

> *Test rn2ncid without connecting it to the ncidd server.*

> > *rn2ncid --test*

> *Choose the Remote Notifier "Send test notification" option.*

Client

- **How many ways does the client display the Caller ID?**

> *The client receives the Caller ID from the server and displays it in one of three ways:*

> > - *Its GUI Window*
> > - *A Terminal Window*
> > - *Using a Output Module*

> *An output module can also send the information to a smart phone, pager, email address and more.*

Client Output Modules

- **What is an output module?**

> *An output module receives the Caller ID information from the ncid client and gives the client new functionality.*

> *There are various output modules than come with NCID and there are also third party ones.*

- **What output modules are available?**

> *The following output modules are distributed:*

> > - *ncid-alert: Send NCID call or message desktop notifications.*
> > - *ncid-initmodem: Reinitialize the modem when "RING" is received as the number.*
> > - *ncid-kpopup: Uses KDE to create a popup for the Caller ID.*
> > - *ncid-mythtv: Displays the Caller ID on MythTV.*
> > - *ncid-notify: Sends the Caller ID on an iOS device or an Android device.*
> > - *ncid-page: Sends the Caller ID to a cell phone or pager.*
> > - *ncid-samba: Creates a popup for the Caller ID on windows using Samba.*
> > - *ncid-skel: Just echos the Caller ID. Modify it to write new modules.*

> - *ncid-speak: Sends the Caller ID to a text-to-speech program.*
> - *ncid-yac: Sends the Caller ID to YAC clients.*
> - *ncid-wakeup: Wakeup X-Windows.*

- **Can output modules be configured?**

> *Output modules are configured using files in the conf.d/ directory. There will be a separate file for each module that needs one, for example, conf.d/**ncid-page.conf**.*

> *For more information, see the comments in the individual files in the conf.d/ directory and the man page for each module.*

- **How are output modules started?**

> *If you are using Debian, Fedora, Ubuntu, FreeBSD, OSX, or raspios you would use the OS specific ncid service script to activate the service.*

> *For distributions not based on Debian, Fedora, or BSD the modules need to be started manually, Here are three examples:*

```
ncid --no-gui --module ncid-page &
ncid --no-gui --module ncid-notify &
ncid --no-gui --module ncid-speak &
```

> *Each of the above commands start ncid using an output module and puts it in the background. The first line starts ncid using the page output module. The second line starts ncid using the notify output module. The third line starts ncid using the speak output module.*

- **How do I configure the page module to send the CID information to my cell phone?**

> *You need to modify one line in **ncid-page.conf** and maybe one line in **ncid.conf**.*

> *Find this line in **ncid.conf**:*

```
PageTo=
```

> *PageTo needs to be set to your mobile provider SMS e-mail address. Here are addresses for the major ones in the US:*

```
Sprint: phonenumber@messaging.sprintpcs.com
Verizon: phonenumber@vtext.com
T-Mobile: phonenumber@tmomail.com
AT&T: phonenumber@txt.att.net
```

> *For example, if your provider is AT&T and your cell number is 1-321-555-1212, then your PageTo line becomes:*
>
> *PageTo=13215551212@txt.att.net*

> *If you want a page anytime the phone rings, you are finished.*

> *if you only want a page if the phone call is unanswered or is at a certain ring count, you need to uncomment the ncid_page line in **ncid.conf**, then change the ring count as desired.*

> *Be careful, all Caller ID devices do not indicate rings. If RING is not sent by the modem, a ring count will not work and the page will never be sent.*

> *If you are using a modem, there is no indication of whether the the phone was answered or not. The modem sends RING to ncidd each time it gets the ringing signal. When RING is not sent anymore ncidd will indicate the end of ringing. A ring count of 4 is a good value to assume the phone was not answered. Remove the '#' so you have this line:*

```
    set ncid_page {set Ring 4}
```

> *If you are using SIP, you can configure it to send the page on hangup without an answer by modifying the above line to:*

```
    set ncid_page {set Ring -1}
```

> *See the comments in the **ncid.conf** file for more information on configuring the ring option line.*

Verbose Levels

> *Table of Contents*

Index

> - *artech2ncid verbose levels*

artech2ncid verbose levels

> *Higher levels include lower levels.*

> *LEVEL9:*

```
    hid_write , X55_counter monitoring ,Ring monitoring
```

> *LEVEL8:*

```
    CID detection and decoding , Hook flash detection
```

> *LEVEL7:*

```
    file and network management (link to ncidd, log file , etc )
```

> *LEVEL6:*

```
    signal handling
```

LEVEL5: dial parsing , Hook flash processing , CID processing

LEVEL4:

```
    incoming/outgoing calls timing (start time  ,pickup time, end_time)
```

LEVEL3:

```
   line state :  ON_LINE ,OFF_LINE ,ON_HOOK,Off_HOOK ,Beep_BEEP , RING
```

LEVEL2:

```
    communication with ncidd
```

LEVEL1:

```
    log command line
    display variable values
    log errors, cleanup, and termination
```

cideasy2ncid verbose levels

Higher levels include lower levels.

LEVEL9:

```
    not used
```

LEVEL8:

```
    not used
```

LEVEL7:

```
    not used
```

LEVEL6:

```
    not used
```

LEVEL5:

```
    hex dump
```

LEVEL4:

```
    received from the cideasy device
    sent to the cideasy device
```

LEVEL3:

```
    indicate ring, on/off hook,
    show Caller ID
```

LEVEL2:

```
    show devices
```

LEVEL1:

```
    log command line
    display variable values
    log errors, cleanup, and termination
```

cidupdate verbose levels

Higher levels include lower levels.

LEVELR:

```
    indicate if failed to open call log
    show rename from and to if asking for Y/n
    show result of attempted rename
```

LEVELE:

```
    indicate if no config file
    indicate if there is a config error
    show system error
    show internal program error
    show Terminated on non-option errors
```

LEVEL9:

```
    not used
```

LEVEL8:

```
    exit normally when position in code reached
```

LEVEL7:

```
    not used
```

LEVEL6:

```
    show line fields created
```

LEVEL5:

```
    not used
```

LEVEL4:

```
    changed blacklist #= name
    changed whitelist #= name
    show old cid line
```

LEVEL3:

```
    show write entry to new call log
```

LEVEL2:

```
    not used
```

LEVEL1:

```
    show start time
    show program name and version
    show command line options
    show config file name
    show alias file name
    show blacklist file name
    show whitelist file name
    show simple expresions or posix or perl regular expressions used
         for alias/blacklist/whitelist entries
    indicate alias file messages
    show if ignoring leading 1 in alias definition and calling number
    indicate blacklist and whitelist file messages
    show if ignoring leading 1 in alias definition and calling number
    show if updating current log or current and previous call logs
    show if skipping interactive prompt
    show end line
    show output call log name
    show terminated [by LEVEL8] with date and time
```

ncid verbose levels

Higher levels include lower levels.

LEVEL9:

```
    not used
```

LEVEL8:

```
    not used
```

LEVEL7:

```
    show getField Default clause
    show inserting row in table list
    show DisplayContextMenu mouse clicks
    show $noteFiles tail of contents, and name
    show country code and NTYPE
```

```
    show putFlag messages
    show tooltip messages
```

LEVEL6:

```
    show $dataBlock for "CID" when sendCID not called
    show label type for $dataBlock
```

LEVEL5:

```
    show show Assigned numbers for data received from ncidd
    show status of host and port at after each config file and command line
```

LEVEL4:

```
    shows fixed font family detected
    shows fixed font skipped
    shows history window font
    shows message window and display font
    shows help text font
    shows top level window geometry
    display history entries in milliseconds
    clock cannot scan start of call time
    clock cannot scan end of call time
    gateway limitations prevent calculation of duration for a CANCEL CALLINFO
```

LEVEL3:

```
    show all discovered lineids
    show attempting to connect
    show server option received
    indicate if no call array label for phone line label
```

LEVEL2:

```
    display history entries in milliseconds
    indicate rcfile and variable changed
    show all server options received
    indicate saved note to $notefile
```

LEVEL1:

```
    indicate if using PID file or not
    display about
    indicate if a output module is being used and which one
    indicate if optional module variable is not being used
    indicate if optional module variable is being used and ring count
    indicate width of name, number, line type and history window
    display received data if in raw mode
    indicate when call log is completely received
    indicate WRK ACCEPTED or REJECTED  to server
    show "GOODBYE"
```

```
    show RELOAD, UPDATE, UPDATES, and REREAD server requests
    show "REQ: PAUSE ..." server requests
    show CIDINFO line on ring count match
    show data sent to module
    show message sent to module
    show CID data sent to module
    show unsupported line labels
    show unsupported alias types
    display popup message
    show discovered lineid's
    show current font
    show number of fixed and total fonts
    show calculated history text field length of message window
    show history text field rows
    indicate window geometry previously not saved or show saved window geometry
    show window geometry
    show window minimum size
    show length of all visible columns in history window
    show number of visible columns out of total number columns
    show resized history field length of message window
    indicate if ncid-alert started in ncid.conf file when autostart is turned on*
    indicate rcfile and variable changed
    indicate unable to save note
    indicate unable to save data in $rcdir because it is a file
    indicate history message and help fonts saved
```

ncidd server verbose levels

> *Higher levels include lower levels.*

LEVELR

```
    indicated received connected signal with date and time
              when a client connects
              if no clients are connected
```

LEVELE

```
    indicate if no config file
    indicate if there is a config error
    indicate if there is no alias file
    indicate if there is a uname system call error
    indicate if open of ncidd log file failed
    indicate if open of call log failed
    indicate warning using noserial and nomodem in config file
    indicate if there is a set TTY error
    indicate warning if no modem or modem busy
    indicate if open TTY port failed
    indicate if TTY port fcntl error
    indicate if TTY port flush error
    indicate warning using normal hangup if using Announce Hangup
             and there is no recording file
```

```
    indicate what the hangup extension must start with "hangup-"
    indicate if hangup extension not found
    indicate if hangup extension must be executable
    indicate warning using normal mode if modem does not support FAX Hangup
    indicate warning using normal hangup if modem does not support
             Announce Hangup
    indicate warning using normal hangup if unknown hangup mode
    indicate if unable to parse ifaddress
    indicate if popen failed for cidupdate
    indicate if pclose failed for cidupdate
    indicate if popen failed for ncidutil
    indicate if pclose failed for ncidutil
    indicate if "sh -c" fails to start hangup script
    indicate if hangup script failed to return hangup or OK
    indicate if pclose failed for hangup script
    indicate status of hangup script after trying to run it
    indicate if TTY port lockfile failed to open
    indicate if failed to remove lockfile
    indicate if signal terminated program
    indicate if  if received unexpected signal
    indicate if system error
    indicate Failed to remove stale lockfile
    indicate if internal program error
    show Terminated with date and time
```

LEVEL9:

```
    (can only be set by the command line)

    show poll() events flag
    show individual tests in strmatch()
    skips LEVEL8
```

LEVEL8:

```
    (can only be set by the command line)

    show alias, blacklist and whitelist tables
    normal exit
```

LEVEL7:

```
    show alias, blacklist and whitelist tables
```

LEVEL6:

```
    indicate client sent empty line
    show optional files failed to open
    indicate sending data to each client
    show progress messages while formatting the phone number
    show progress messages while interpreting the MESG
```

LEVEL5:

```
    display data as a hexdump
    show modem command return codes for hangup
    show lastring and ring count if ring detected
    show processing hangup request
    show reason for not sending to serial device in initModem
```

LEVEL4:

```
    indicate if name and nmbr received but no date or time
    indicate date, time, nmbr received
    indicate date, time, name or date, time, name, mesg received
    indicate date, time, nmbr, mesg received
    indicate date, time, nmbr, name or date, time, nmbr, name, mesg received
    detected TCI serial device format
    detected NetCallerID or gateway format
    indicate number without RING is Call Waiting (WID)
    show alias matching Begin: and End: with time and result
    show open, Begin, End and exit status for external cidupdate, ncidutil, and hangup
scripts
```

LEVEL3:

```
    show number of tries to init modem
    show modem responses
    show modem query commands for software version, country code, operation modes
    indicate Non 7-bit ASCII message deleted
    indicate Gateway sent CALL data
    indicate Gateway sent CALLINFO data
    indicate client sent text message
    indicate client sent unknown data
    show call data input
    show CID line
    show ACK line
    show calltype, hangup, hupmode, cidline and lineid
    if true, indicate cidline == lineid: check whitelist
    if true, indicate name of number on whitelist: skip hangup check
    if true, show cidline != lineid: skip hangup check
    show checked whitelist and blacklist for match or whitelist empty
    indicate match in blacklist or whitelist with values
    show hangup extension and arguments
    show default mode for hangup extension
    show using hangup mode
    show using recording
    show hangup extension return code
    show meaning of MESG code if known and discarded
    indicate Removed stale lockfile
    show unavailable modems during hangup processing
```

LEVEL2:

```
    show lines that start with a 3 digit number
    show client connect/disconnect
    indicate if client sent a HELLO Identification line
    indicate if client sent a HELLO Command line
    indicate if client command accepted
    indicate end of client hello lines
    indicate status of call log
    indicate if LineIDS sent to client
    indicate if server options sent to client
    indicate end of cobnnection startup
    indicate hangup pause, hangup not paused, or hangup enabled
    show number of times socket was zero in sequence when trying to remove client from
poll
    show client removed on write error
    show client not found in poll table
    indicate network client hung up
    indicate device or modem returned no data
    indicate end of call log or call log empty or no call log sent
    indicate end of connection startup
    show line sent to cidupdate
    show line sent to ncidutil
    show OPT lines
    show REQ: lines
    show INFO: lines
    show WRK: lines
    show RESP: lines
    indicate case where finishCID did not receive full call info
```

> *LEVEL1:*

```
    show started with date and time
    show server version
    indicate if could not create ncidd logfile
    indicate name and location of ncidd logfile
    indicate if no config file or config file processed
    indicate set command skipped in config file
    indicate error in opening ncidd log file
    indicate what is configured to send to the clients
    show verbose level
    indicate data type sent to clients
    indicate alias file messages
    indicate if leading 1 needed for aliases
    indicate blacklist  and whitelist file messages
    indicate alias, blacklist and whitelist total/maximum entries, if any
    indicate cid logfile messages
    indicate of no call logfile
    indicate name and location of call logfile
    indicate if no data logfile
    indicate name and location of data logfile
    show Telephone Line Identifier
    show TTY port opened
    show TTY port speed
```

```
    show name and location of TTY lockfile
    show modem query results
    indicate TTY port control signals enabled or disabled
    indicate Caller ID from a serial or USB device and optional gateways
    indicate Caller ID from a modem and optional gateways
    indicate Handles modem calls without Caller ID
    indicate Does not handle modem calls without Caller ID
    indicate Caller ID from gateways without modem support
    indicate hangup option
    show network port
    indicate not using PID file if no '-P' option
    indicate pid file already exists
    indicate found stale pidfile
    indicate cannot write pidfile
    indicate wrote pid in pidfile
    indicate end of startup
    indicate TTY in use with date and time
    indicate TTY free with date and time
    indicate cannot init TTY and terminated with date and time
    indicate Modem initialized.
    indicate Initialization string for modem is null.
    indicate Modem Chipset version
    indicate Modem set for CallerID.
    indicate Modem set for CallerID and Call Waiting.
    indicate Modem does or does not support FAX
    indicate Modem supports Snooping for call waiting
    indicate CallerID initialization string for modem is null.
    indicate CallerID TTY port initialized
    indicate serial device hung up and terminated with date and time
    indicate device error and terminated with date and time
    indicate serial device error and terminated with date and time
    indicate poll error
    indicate invalid request from serial device, terminated with date and time
    indicate Invalid Request, removed client
    indicate Write event not configured, removed client
    indicate device or modem read error
    indicate Device returns no data, Terminated with date and time
    indicate network connect error
    indicate network NOBLOCK error
    indicate too many network clients
    indicate network client read error
    indicate cid log is too large
    indicate sending log to client
    indicate removed pidfile
    indicate signal received and terminate program with date and time
    indicate SIGHUP received and reload alias files
    indicate SIGPIPE received and ignored with date and time
    indicate success or failure to parse ifaddr to set the interface
    indicate which country code is set for number formatting
    show number display format selected if country is NANPA
    show error in starting external cidupdate, ncidutil, and hangup scripts
    indicate hangup script did not return hangup or OK
```

```
    indicate unable to go off hook to do the hangup
    indicate trouble in going on hook to finish the hangup
    indicate no modem found for cidline - skipped hangup
    show the message text sent to clients by sysMesg
```

ncid2ncid gateway verbose levels

> *Higher levels include lower levels.*

> *LEVELE*

```
    indicate if no config file
    indicate if there is a config error
    show gethostname error
```

> *LEVEL9:*

```
    (can only be set by the command line)
    not used
```

> *LEVEL8:*

```
    (can only be set by the command line)
    not used
```

> *LEVEL7:*

```
    not used
```

> *LEVEL6:*

```
    not used
```

> *LEVEL5:*

```
    not used
```

> *LEVEL4:*

```
    indicate reading socket
    show call log lines received
```

> *LEVEL3:*

```
    show all data received from all servers
```

> *LEVEL2:*

```
    indicate line sent to receiving NCID server
    indicate line from receiving NCID server
```

> *LEVEL1:*

```
    indicate cannot create or open existing logfile
    show start date and time
    show server version
    show command line
    indicate Debug mode
    indicate no config file or config file processed
    indicate set command skipped in config file
    show HELLO: IDENT:
    show HELLO: CMD:
    show error line in config file
    show verbose level
    indicate not using PID file, there was no '-P' option
    indicate found stale pidfile
    indicate wrote pid in pidfile
    show Receiving Host host:port
    show server greeting line
    show configured Sending Hosts host:port
    show configured servers greeting line
    indicate client disconnected
    indicate client reconnected
    indicate Hung Up
    indicate Poll Error
    indicate Removed client, invalid request
    indicate Removed client, write event not configured
    indicate line cannot be sent to receiving NCID server
    indicate removed pidfile
    show terminated with date and time
```

ncidnumberinfo verbose levels

> *Higher levels include lower levels.*

> *LEVELE:*

```
    indicate if a phone number must be given
    show usage for adding a phonenumber
    show system error
    show internal program error
    show Terminated on non-option errors
```

> *LEVEL9:*

```
    not used
```

> *LEVEL8:*

```
    exit normally when position in code reached
```

> *LEVEL7:*

```
    not used
```

> *LEVEL6:*

```
    not used
```

> *LEVEL5:*

```
    not used
```

> *LEVEL4:*

```
    not used
```

> *LEVEL3:*

```
    show raw telephone number
```

> *LEVEL2:*

```
    not used
```

> *LEVEL1:*

```
    show start time
    show program name and version
    show command line options
    show verbose level
    indicate NANP country format used
    show terminated [by LEVEL8] with date and time
```

obi2ncid gateway verbose levels

> *Higher levels include lower levels.*

> *LEVEL9:*

```
    not used
```

> *LEVEL8:*

```
    not used
```

> *LEVEL7:*

```
    show call log from ncidd, if received
```

> *LEVEL6:*

```
    indicate start and end of received packets
    filter received packets
```

> *LEVEL5:*

```
    show call or message line from ncidd
    show log lines received from the obi
```

LEVEL4:

```
    show what matched on a log line from the obi
    show variables set by the match
```

LEVEL3:

```
    show CALL: line generated
    show CALLINFO: line generated
    indicate Outgoing call not completed
```

LEVEL2:

```
    not used
```

LEVEL1:

```
    show Started
    show command line and any options on separate lines
    show logfile name and opened as append or overwrite or could not open
    show processed config file or config file not found
    show name and version
    show verbose level
    show Hostname flag
    show IDENT
    show Command
    show Line ID
    show NCID address:port
    show delay time between retrying failed connection
    show debug mode if in debug mode
    show test mode if in test mode
    show PID or some PID problem or not using PID file
    show connected to NCID <address:port> or error exit
    show greeting line from NCID
    show listening port or error exit
    show exit on error
    show signals ingnored
    show NCID server disconnected if it goes away and trying to reconnect
    show terminated and signal that caused it
```

rn2ncid gateway verbose levels

Higher levels include lower levels.

LEVEL9:

```
    not used
```

LEVEL8:

```
    not used
```

LEVEL7:

```
    not used
```

LEVEL6:

```
    not used
```

LEVEL5:

```
    show call log from ncidd, if received
    show call or message line from ncidd
    show messgae line from remote notification
```

LEVEL4:

```
    not used
```

LEVEL3:

```
    show notification type
    show Call: line generated if type RING
    show NOT: line generated if type PING, Battery, SMS, MMS, or VOICEMAIL
    show notice of a SMS or MMS message
    show unknown notification type
    show notification type was rejected
```

LEVEL2:

```
    not used
```

LEVEL1:

```
    show Started
    show command line and any options on separate lines
    show logfile name and opened as append or overwrite or could not open
    show processed config file or config file not found
    show name and version
    show verbose level
    show HELLO: IDENT:
    show HELLO: CMD:
    show Line ID
    show debug mode if in debug mode
    show test mode if in test mode
    show reject option values or none
    show pid or some PID problem
    show connected to NCID <address:port> or error exit
    show greeting line from NCID
    show delay between each try to reconnect to server
```

```
    show listening port or error exit
    show NCID server disconnected if it goes away and trying to reconnect
```

sip2ncid gateway verbose levels

> *Higher levels include lower levels.*

> *LEVEL9:*

```
    (can only be set by the command line)

    show lines received from the NCID server
```

> *LEVEL8:*

```
    not used
```

> *LEVEL7:*

```
    not used
```

> *LEVEL6:*

```
    not used
```

> *LEVEL5:*

```
    not used (reserved for hex dumps)
```

> *LEVEL4:*

```
    show startup lines from server
    show packet from and to addresses
    show packet source and destination ports
    show packet data size in bytes
    show linenum array and contact as they are compared for an INVITE
    indicate checked for outgoing call
    show INVITE contact was not registered for out call
    indicate Alarm Timeout and msgsent flag
    show call log if sent
    show Loopback encapsulation type
```

> *LEVEL3:*

```
    show SIP packets
    give character count of lines received from the NCID server
    show protocol information for the packet
    show warning SIP packet truncated
    indicate examining packet for line label
    indicate number, or name and number, in packet
    show alarm timeout, pcap\_loop() return value and msgsent flags
    show calls table search, additions and deletions
```

LEVEL2:

```
show CALL and CALLINFO lines
show packet number received and date
show request line
show outgoing call
show cidmsg log line generated
show trying responses
show Warning: could not connect to the NCID server
```

LEVEL1:

```
indicate cannot create or open existing logfile
show start date and time
show server and API versions
indicate test mode
indicate Debug mode
indicate no config file
indicate config file processed
indicate set command skipped in config file
show error line in config file
indicate Reading from dumpfile
indicate Writing to dumpfile
show verbose level
show HELLO: IDENT:
show HELLO: CMD:
show Line ID
show status: Warn clients: 'No SIP packets' & 'SIP packets returned'
show NCID status
show network interface used
show applied filter
indicate no filter applied
indicate No SIP packets received
indicate SIP packets returned
indicate not using PID file, there was no '-P' option
indicate pid file already exists
indicate found stale pidfile
indicate wrote pid in pidfile
alarm SIP packets returned
Warning: SIP Packet truncated
Warning: simultaneous calls exceeded
invalid IP header length
show registered line number
indicate Number of telephone lines exceeded
show CID line sent to NCID
indicate packet parse problems
indicate caller hangup before answer
indicate hangup after answer
Warning: cannot get CallID
Warning: Warning no SIP packets
indicate pcap_loop error
indicate removed pidfile
```

```
    indicate program terminated with date and time
    show extension size
    show pcap linktype for the device
```

wc2ncid gateway verbose levels

> *Higher levels include lower levels.*

> *LEVEL9:*

```
    not used
```

> *LEVEL8:*

```
    not used
```

> *LEVEL7:*

```
    not used
```

> *LEVEL6:*

```
    not used
```

> *LEVEL5:*

```
    show call log from ncidd, if received
    show Caller ID line from ncidd
```

> *LEVEL4:*

```
    show hex dump of received packet
```

> *LEVEL3:*

```
    show unit and serial numbers from Whozz Calling device
    show Call line from Whozz Calling device
```

> *LEVEL2:*

```
    show CALL and CALLINFO lines sent to ncidd
    show Phone Off Hook
    show Phone On Hook
```

> *LEVEL1:*

```
    show Started
    show command line and any options on separate lines
    show name and version
    show verbose level
    show debug mode if in debug mode
    show test mode if in test mode
    show logfile name and whether opened as append or overwrite
```

```
    show logfile could not be opened
    show processed config file or config file not found
    show Trying to connect to <ipaddr>:<port>
    show Connected to NCID server at <ipaddr>:<port
    show Hostname flag
    show IDENT
    show discovered WV devices if more than 1
    show Device WC-<number> at address
    show Sent ^^Id-Z<mmddhhmm>" to <IPaddress>:<port> to set internal clock
    show Command
    show connected to NCID <address:port> or error exit
    show greeting line from NCID
    show opened broadcast port
    show closed broadcast port
    show opened WC device port
    show closed WC device port
    show commands sent
    show Pause after sending ^^Id-V
    show checking and setting required flags
    indicate command data received or timeout in seconds
    show data from some commands
    show Waiting for calls from <server:port>
```

xdmf2ncid gateway verbose levels

> *Higher levels include lower levels.*

> *LEVEL9:*

```
    not used
```

> *LEVEL8:*

```
    not used
```

> *LEVEL7:*

```
    not used
```

> *LEVEL6:*

```
    indicate start and end of received packets
    filter received packets
```

> *LEVEL5:*

```
    show call log from ncidd, if received
    show call or message line from ncidd
```

> *LEVEL4:*

```
    indicate U counts from Holtek HT9032D
    show Hex Dump of Message
    indicate <SDMF|XDMF> packet and bytes to checksum
    show Message Length
    show Calculated Checksum Good
    show Got Call Type
    show Got Date & Time
    show Got Caller Number
    show Got Why No Number
    show Got Caller Name
    show Got Why No Name
    show Got unknown
    show Detected Call type
```

LEVEL3:

```
    indicate data received from Holtek HT9032D
    indicate Ignored Holtek HT9032D noise packet
    indicate detected Comet device
    indicate ASCII Hex detected
    indicate Detected XDMF Hex message
    show Received Message
    show Calculated Checksum Bad, if bad
    show Call line sent to server
    indicate ERROR - not an XDMF packet
```

LEVEL2:

```
    not used
```

LEVEL1:

```
    show Started
    show command line and any options on separate lines
    show logfile name and opened as append or overwrite or could not open
    show processed config file or config file not found
    show name and version
    indicate configured for Holtek HT9032D
    indicate configured for a Comet or modem
    show verbose level
    show Hostname Flag
    show HELLO: IDENT:
    show HELLO: CMD:
    show Line ID
    show NCID <address:port>
    show delay time between retrying failed connection
    show <ttyport>
    show debug mode if in debug mode
    show test mode if in test mode
    show PID or some PID problem or not using PID file
    show connected to NCID <address:port> or error exit
    show greeting line from NCID
```

```
    show Connected to USB port <ttyport>
    show Waiting for calls from <ttyport>
    show exit on error
    show signals ingnored
    show NCID server disconnected if it goes away and trying to reconnect
    show terminated and signal that caused it
```

NCID Contributors

> *Table of Contents*

> *Any omissions are entirely my fault. Please notify jlc of any corrections or additions.*

John L. Chmielewski

- Designed, developed and wrote most of the programs.

Mark Salyzyn

- Ported `ncidd` to BSD and Macintosh.
- Wrote `getopt.c` and `poll.c`.

Mace Moneta

- Wrote `nciduser`, which was the basis for `ncid-speak`.
- Contributed ideas and code to `ncid` client.
- Contributed the Mac OS X portion of `ncid-speak`.

Dan Lawrence

- Contributed to `ncid-email` so paging would work.
- Contributed information on freewrap for `ncid` client on Windows.

Aron Green

- Helped fix `termios` settings to work with FreeBSD.
- Contributed `ncid.sh` and `ncidd.sh` start/stop scripts for FreeBSD.
- Contributed ideas for `ncidd` and `ncid`.

Troy Carpenter

- Developed `ncid-samba` to send CID info to Samba for a Windows popup.

Lyman Epp

- Wrote the first version of `ncidrotate` in python.

Rick Matthews

- Provided information on distinctive ring.

Michael Nygren

- Provided information on the +GCI modem code, so CID will work with modems that need a country code.

Mitch Riley

- Provided information needed to create the `ncid-mythtv` script.

Roger Knobber

- Provided patch for *strdate()* in `ncidd.c` to fix null pointer in *gettimeofday()* in version 0.61.

Rich West

- Helps maintain the `ncid-mythtv` module.
- Provided an NSIS script as a basic installer for the `ncid` Windows client.

Clayton O'Neill

- Modified `ncidd` to be able to run with no serial device.
- Added the ability to inject CID from clients.
- Contributed the `ncid-sipinject` program which was renamed `ncidsip`.

David LaPorte

- Improved `ncidsip` to work with a missing name.
- Improved `ncidsip` to detect outgoing calls containing a SIP REGISTER packet so they are not treated as incoming calls.

Michael Lasevich

- Wrote and contributed `yac2ncid`.
- `ncid-yac` was developed from a module he wrote.
- Helped write the man pages.

Randy Rasmussen

- Wrote and contributed the `ncid-kpopup` client output module.

Jonathan Wolf

- Hacked `ncid` to provide ring indication when Caller ID not available. (His hack was not used, but his feature was added to `ncidd`.)

littlepepper

- Provided the Mac OS X modem init string for the iMac.

Marko Koski-Vähälä

- Contributed `ncid` client code to format numbers for Sweden. The client can now format numbers for "US" and "SE".

Chris Lenderman

- Helped rewrite *sendLog()* code to eliminate corruption sending large log files (greater than 2,000 lines).
- Converted `testclient` from using *netcat* to using sockets.
- Maintained the Windows version of `NCIDPop`.
- Created (and maintains) a complete rewrite of `NCIDPop` in Java making `NCIDPop` cross-platform for Mac, Windows, Linux.
    - `Created (and maintains) `NCID Android` client.`

Todd Andrews

- Corrected problems in the INSTALL document.

- Helped with server testing and fixing various Mac OS X problems.
- Provided a server fix for the hangup configuration/option problem on Mac OS X.
- Developed `ncid-nma` output module which became part of `ncid-notify`.
- Developed `wc2ncid` gateway.
- Helped improve various NCID documentation.
- Did a lot of work for the NCID 0.85 and 0.86 distributions.
- In general, helps with various NCID programs and documentation.
- provided help with verious NCID issues through the years

Jeff Rabin

- Helped with the `ncidd` hangup option by testing, reviewing documentation, suggesting improvements and providing patches.
- Created the `ncidd` configuration option *ignore1* patches.

[Neven Ćosić (Sensei)](mailto:senseitcom (at) email (dot) t (hyphen) com (dot) hr)

- Added country code HR (Croatia) to `ncid` and `ncid.conf`.
- Added alternate date formats and separators.
- Wrote a `ncid-tiny` module that became the basis for `ncid-alert`.
- Introduced and tested the CTI Comet USB Caller ID device.
- Introduced and tested the udev rules.
- Added CallLog option to the NCID client for the NoGUI mode.
- Added NightMode option to the NCID client for the GUI mode.
- Wrote NCID Alert Autostart file that became the basis for a modified version that is generated from the ncid.desktop file when needed.
- Added `udev-action` auxiliary script.
- Introduced and edited the documentation in PDF format.
- Added alert_call and alert_message options to `ncid-alert`, makes Caller ID and Integral Message or Notice data shown in Alert messages configurable.
- Added YearDot option to the NCID client.
- Introduced and tested the Holtek HT9032D based PSTN Caller ID module.
- In general, helps with various NCID programs and documentation.

Steve Limkemann

- Modified `ncidd` to output CID information quickly if a name is not part of the Caller ID received.
- Improved the `cidupdate` tool.
- Improved the `ncid` GUI, by adding multiple features such as window resizing and the ability to change font names and font sizes.
- Added wakeup feature to `ncid` and wrote the `ncid-wakeup` module.
- In general, helps with various NCID programs.

Tod Cox

- Ported NCID to the Raspberry Pi running the Raspbian OS.

Nicholas Riley

- Maintained `NCIDPop` for Mac OS X.

Alexei Kosut

- Original developer of `NCIDPop` for Mac OS X and Windows.

David J. Lauria

- developed ncid-fly for the TiVo
- helped with ncidmod for the TiVo
- provided NCID compatibility into his cidrss Tivoweb module
- provided help with verious NCID issues through the years

Reece Pollack

- Provided patch to ncidd for regular expressions

Kris Jensen

- developed procedure for creating the announce.rmd file.
- donated modem recording files:
  NumberDisconnected.wav and NumberDisconnected.rmd

Pete Bekatoros

- donated modem recording files:
  CallingDeposit.pvf CallingDeposit.rmd
  CannotBeCompleted.pvf CannotBeCompleted.rmd
  NotInService.pvf NotInService.rmd

Mike Stember

- Developed a way to process the United States Federal Communications Commission (FCC) robocaller/telemarketer data file in Excel, filter out the most likely candidates and create a text file that is NCID-friendly.
- Volunteered to monitor the FCC site for updates and continues to maintain the NCID-friendly text file as a download on the NCID Wiki.

Randy Tarantino

- created ncid-mysql output module to log call data to MySQL database

Reece

- fixed voice hangup to work with USB modems
- see bug #25

Ed Attfield

- wrote part of the DIAL code
- wrote hangup-fakenum
- updated [FCC Data on Unwanted Calls](FCC2ncid at https://sourceforge.net/p/ncid/wiki/FCC%20Data%20on%20Unwanted%20Calls/)
- rewrote saveSend() in ncidd.c
- cleaned up ncidd.c

Jason Trinklein

- provided steps to use Linphone as a SIP client

Jon Tulk

- created the hangup-greylist extension and its man pages

Bruno Grasland

- submitted bug reports and fixes for ncidd, ncid-mythtv and ncidutil
- implemented WID (call waiting) for modems that support it

- Proposed and implemented integration of libphonenumber in nicd for international number formating and displaying of phone numbers geographical data.
- wrote libcarrier (c++ derivative of libgeocoding) in order to be able to display phone number carriers in ncid
- wrote the original scripts for carrier metadata generation for US/CAN/FR/JP

New Feature Requests, Bug Reports, Testing Fixes and Testing New features

> *Adam 'Starblazer' Romberg*
> *Andy Nunez*
> *Andy Writter*
> *Aron Green*
> *Carl Johnson*
> *Charlie Heitzig*
> *Dan Lawrence*
> *David LaPorte*
> *George Johnson*
> *Joe Nardone*
> *Jonathan Wolf*
> *Ken Appell*
> *Lloyd Stahlbush*
> *Mace Moneta*
> *Marko Koski-Vähälä*
> *Matt Short*
> *Michael Lasevich*
> *Nicholas Davies*
> *Paul Miller*
> *Phil Fitzpatrick*
> *Rick Matthews*
> *Steve Forman*
> *Steve Major*
> *Troy Carpenter*

Feedback On Working Modems

> *Derek Huxley*

TODO

> *Table of Contents*

TODO Lists

> - *TODO*

- Maybe

The TODO list (in no particular order)

> - *Add ncid.exe to the Windows tray*
> - *Add a GUI configuration program for the server, client, modules and gateways*
> - *Create a Mac OS X installer*
> - *Allow wc2ncid/wct to configure devices out-of-subnet*

- *Add hangup support to gateways that can hangup a call*
- *post a NEWS item for INSTALL-Mac when MacPorts fixes hidapi package that currently does not install hidapi_darwin.h*

- *Make line terminator usage consistent between server, clients, gateways (<CR>, <LF>, <NL>)*

The Maybe list (in no particular order)

- *Add blacklist/whitelist support to wc2ncid gateway to block calls*
- *Add support for PBX*
- *Add an iax2 gateway*
- *Add ability for client to get name from address book.*
- *Add ringtone support (NCIDpop and ncid-applet have ringtone support)*
- *Add a Firefox module*
- *Add SSL between server and client for secure use over Internet*
- *Use separate ports for clients and gateways*
- *Add SIP blocking and recording capability*
- *Add a Modem On Hold feature or at least display the Caller ID during a modem data call*
- *Add XBMC support*
- *Add graphics to view image of who is calling*
- *Allow Android gateway to tell an Android phone to hangup*
- *Redimensionable columns in NCID client*
- *Display voicemail state (# of mesg) in separate widget in NCID client*

GNU GENERAL PUBLIC LICENSE

Version 3, 29 June 2007

Preamble

The GNU General Public License is a free, copyleft license for software and other kinds of works.

The licenses for most software and other practical works are designed to take away your freedom to share and change the works. By contrast, the GNU General Public License is intended to guarantee your freedom to share and change all versions of a program--to make sure it remains free software for all its users. We, the Free Software Foundation, use the GNU General Public License for most of our software; it applies also to any other work released this way by its authors. You can apply it to your programs, too.

When we speak of free software, we are referring to freedom, not price. Our General Public Licenses are designed to make sure that you have the freedom to distribute copies of free software (and charge for them if you wish), that you receive source code or can get it if you want it, that you can change the software or use pieces of it in new free programs, and that you know you can do these things.

To protect your rights, we need to prevent others from denying you these rights or asking you to surrender the rights. Therefore, you have certain responsibilities if you distribute copies of the software, or if you modify

it: responsibilities to respect the freedom of others.

For example, if you distribute copies of such a program, whether gratis or for a fee, you must pass on to the recipients the same freedoms that you received. You must make sure that they, too, receive or can get the source code. And you must show them these terms so they know their rights.

Developers that use the GNU GPL protect your rights with two steps:

1. assert copyright on the software, and
2. offer you this License giving you legal permission to copy, distribute and/or modify it.

For the developers' and authors' protection, the GPL clearly explains that there is no warranty for this free software. For both users' and authors' sake, the GPL requires that modified versions be marked as changed, so that their problems will not be attributed erroneously to authors of previous versions.

Some devices are designed to deny users access to install or run modified versions of the software inside them, although the manufacturer can do so. This is fundamentally incompatible with the aim of protecting users' freedom to change the software. The systematic pattern of such abuse occurs in the area of products for individuals to use, which is precisely where it is most unacceptable. Therefore, we have designed this version of the GPL to prohibit the practice for those products. If such problems arise substantially in other domains, we stand ready to extend this provision to those domains in future versions of the GPL, as needed to protect the freedom of users.

Finally, every program is threatened constantly by software patents. States should not allow patents to restrict development and use of software on general-purpose computers, but in those that do, we wish to avoid the special danger that patents applied to a free program could make it effectively proprietary. To prevent this, the GPL assures that patents cannot be used to render the program non-free.

The precise terms and conditions for copying, distribution and modification follow.

TERMS AND CONDITIONS

0. Definitions.

This License refers to version 3 of the GNU General Public License.

Copyright also means copyright-like laws that apply to other kinds of works, such as semiconductor masks.

The Program refers to any copyrightable work licensed under this License. Each licensee is addressed as *you*. *Licensees* and *recipients* may be individuals or organizations.

To *modify* a work means to copy from or adapt all or part of the work in a fashion requiring copyright permission, other than the making of an exact copy. The resulting work is called a *modified version* of the earlier work or a work *based on* the earlier work.

A *covered work* means either the unmodified Program or a work based on the Program.

To *propagate* a work means to do anything with it that, without permission, would make you directly or secondarily liable for infringement under applicable copyright law, except executing it on a computer or modifying a private copy. Propagation includes copying, distribution (with or without modification), making available to the public, and in some countries other activities as well.

To *convey* a work means any kind of propagation that enables other parties to make or receive copies. Mere interaction with a user through a computer network, with no transfer of a copy, is not conveying.

An interactive user interface displays *Appropriate Legal Notices* to the extent that it includes a convenient and prominently visible feature that

1. displays an appropriate copyright notice, and
2. tells the user that there is no warranty for the work (except to the extent that warranties are provided), that licensees may convey the work under this License, and how to view a copy of this License.

If the interface presents a list of user commands or options, such as a menu, a prominent item in the list meets this criterion.

1. Source Code.

The *source code* for a work means the preferred form of the work for making modifications to it. *Object code* means any non-source form of a work.

A *Standard Interface* means an interface that either is an official standard defined by a recognized standards body, or, in the case of interfaces specified for a particular programming language, one that is widely used among developers working in that language.

The *System Libraries* of an executable work include anything, other than the work as a whole, that (a) is included in the normal form of packaging a Major Component, but which is not part of that Major Component, and (b) serves only to enable use of the work with that Major Component, or to implement a Standard Interface for which an implementation is available to the public in source code form. A *Major Component*, in this context, means a major essential component (kernel, window system, and so on) of the specific operating system (if any) on which the executable work runs, or a compiler used to produce the work, or an object code interpreter used to run it.

The *Corresponding Source* for a work in object code form means all the source code needed to generate, install, and (for an executable work) run the object code and to modify the work, including scripts to control those activities. However, it does not include the work's System Libraries, or general-purpose tools or generally available free programs which are used unmodified in performing those activities but which are not part of the work. For example, Corresponding Source includes interface definition files associated with source files for the work, and the source code for shared libraries and dynamically linked subprograms that the work is specifically designed to require, such as by intimate data communication or control flow between those subprograms and other parts of the work.

The Corresponding Source need not include anything that users can regenerate automatically from other parts of the Corresponding Source.

The Corresponding Source for a work in source code form is that same work.

2. Basic Permissions.

All rights granted under this License are granted for the term of copyright on the Program, and are irrevocable provided the stated conditions are met. This License explicitly affirms your unlimited permission to run the unmodified Program. The output from running a covered work is covered by this License only if the output, given its content, constitutes a covered work. This License acknowledges your rights of fair use or other equivalent, as provided by copyright law.

You may make, run and propagate covered works that you do not convey, without conditions so long as your license otherwise remains in force. You may convey covered works to others for the sole purpose of having them make modifications exclusively for you, or provide you with facilities for running those works, provided that you comply with the terms of this License in conveying all material for which you do not control copyright. Those thus making or running the covered works for you must do so exclusively on your behalf,

under your direction and control, on terms that prohibit them from making any copies of your copyrighted material outside their relationship with you.

Conveying under any other circumstances is permitted solely under the conditions stated below. Sublicensing is not allowed; section 10 makes it unnecessary.

3. Protecting Users' Legal Rights From Anti-Circumvention Law.

No covered work shall be deemed part of an effective technological measure under any applicable law fulfilling obligations under article 11 of the WIPO copyright treaty adopted on 20 December 1996, or similar laws prohibiting or restricting circumvention of such measures.

When you convey a covered work, you waive any legal power to forbid circumvention of technological measures to the extent such circumvention is effected by exercising rights under this License with respect to the covered work, and you disclaim any intention to limit operation or modification of the work as a means of enforcing, against the work's users, your or third parties' legal rights to forbid circumvention of technological measures.

4. Conveying Verbatim Copies.

You may convey verbatim copies of the Program's source code as you receive it, in any medium, provided that you conspicuously and appropriately publish on each copy an appropriate copyright notice; keep intact all notices stating that this License and any non-permissive terms added in accord with section 7 apply to the code; keep intact all notices of the absence of any warranty; and give all recipients a copy of this License along with the Program.

You may charge any price or no price for each copy that you convey, and you may offer support or warranty protection for a fee.

5. Conveying Modified Source Versions.

You may convey a work based on the Program, or the modifications to produce it from the Program, in the form of source code under the terms of section 4, provided that you also meet all of these conditions:

- a) The work must carry prominent notices stating that you modified it, and giving a relevant date.
- b) The work must carry prominent notices stating that it is released under this License and any conditions added under section 7. This requirement modifies the requirement in section 4 to *keep intact all notices*.
- c) You must license the entire work, as a whole, under this License to anyone who comes into possession of a copy. This License will therefore apply, along with any applicable section 7 additional terms, to the whole of the work, and all its parts, regardless of how they are packaged. This License gives no permission to license the work in any other way, but it does not invalidate such permission if you have separately received it.
- d) If the work has interactive user interfaces, each must display Appropriate Legal Notices; however, if the Program has interactive interfaces that do not display Appropriate Legal Notices, your work need not make them do so.

A compilation of a covered work with other separate and independent works, which are not by their nature extensions of the covered work, and which are not combined with it such as to form a larger program, in or on a volume of a storage or distribution medium, is called an *aggregate* if the compilation and its resulting copyright are not used to limit the access or legal rights of the compilation's users beyond what the individual works permit. Inclusion of a covered work in an aggregate does not cause this License to apply to the other parts of the aggregate.

6. Conveying Non-Source Forms.

You may convey a covered work in object code form under the terms of sections 4 and 5, provided that you also convey the machine-readable Corresponding Source under the terms of this License, in one of these ways:

- a) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by the Corresponding Source fixed on a durable physical medium customarily used for software interchange.
- b) Convey the object code in, or embodied in, a physical product (including a physical distribution medium), accompanied by a written offer, valid for at least three years and valid for as long as you offer spare parts or customer support for that product model, to give anyone who possesses the object code either
    1. a copy of the Corresponding Source for all the software in the product that is covered by this License, on a durable physical medium customarily used for software interchange, for a price no more than your reasonable cost of physically performing this conveying of source, or
    2. access to copy the Corresponding Source from a network server at no charge.
- c) Convey individual copies of the object code with a copy of the written offer to provide the Corresponding Source. This alternative is allowed only occasionally and noncommercially, and only if you received the object code with such an offer, in accord with subsection 6b.
- d) Convey the object code by offering access from a designated place (gratis or for a charge), and offer equivalent access to the Corresponding Source in the same way through the same place at no further charge. You need not require recipients to copy the Corresponding Source along with the object code. If the place to copy the object code is a network server, the Corresponding Source may be on a different server operated by you or a third party) that supports equivalent copying facilities, provided you maintain clear directions next to the object code saying where to find the Corresponding Source. Regardless of what server hosts the Corresponding Source, you remain obligated to ensure that it is available for as long as needed to satisfy these requirements.
- e) Convey the object code using peer-to-peer transmission, provided you inform other peers where the object code and Corresponding Source of the work are being offered to the general public at no charge under subsection 6d.

A separable portion of the object code, whose source code is excluded from the Corresponding Source as a System Library, need not be included in conveying the object code work.

A *User Product* is either

1. a *consumer product*, which means any tangible personal property which is normally used for personal, family, or household purposes, or
2. anything designed or sold for incorporation into a dwelling.

In determining whether a product is a consumer product, doubtful cases shall be resolved in favor of coverage. For a particular product received by a particular user, *normally used* refers to a typical or common use of that class of product, regardless of the status of the particular user or of the way in which the particular user actually uses, or expects or is expected to use, the product. A product is a consumer product regardless of whether the product has substantial commercial, industrial or non-consumer uses, unless such uses represent the only significant mode of use of the product.

Installation Information for a User Product means any methods, procedures, authorization keys, or other information required to install and execute modified versions of a covered work in that User Product from a modified version of its Corresponding Source. The information must suffice to ensure that the continued functioning of the modified object code is in no case prevented or interfered with solely because modification has been made.

If you convey an object code work under this section in, or with, or specifically for use in, a User Product, and the conveying occurs as part of a transaction in which the right of possession and use of the User Product is transferred to the recipient in perpetuity or for a fixed term (regardless of how the transaction is characterized), the Corresponding Source conveyed under this section must be accompanied by the Installation Information. But this requirement does not apply if neither you nor any third party retains the ability to install modified object code on the User Product (for example, the work has been installed in ROM).

The requirement to provide Installation Information does not include a requirement to continue to provide support service, warranty, or updates for a work that has been modified or installed by the recipient, or for the User Product in which it has been modified or installed. Access to a network may be denied when the modification itself materially and adversely affects the operation of the network or violates the rules and protocols for communication across the network.

Corresponding Source conveyed, and Installation Information provided, in accord with this section must be in a format that is publicly documented (and with an implementation available to the public in source code form), and must require no special password or key for unpacking, reading or copying.

7. Additional Terms.

Additional permissions are terms that supplement the terms of this License by making exceptions from one or more of its conditions. Additional permissions that are applicable to the entire Program shall be treated as though they were included in this License, to the extent that they are valid under applicable law. If additional permissions apply only to part of the Program, that part may be used separately under those permissions, but the entire Program remains governed by this License without regard to the additional permissions.

When you convey a copy of a covered work, you may at your option remove any additional permissions from that copy, or from any part of it. (Additional permissions may be written to require their own removal in certain cases when you modify the work.) You may place additional permissions on material, added by you to a covered work, for which you have or can give appropriate copyright permission.

Notwithstanding any other provision of this License, for material you add to a covered work, you may (if authorized by the copyright holders of that material) supplement the terms of this License with terms:

- a) Disclaiming warranty or limiting liability differently from the terms of sections 15 and 16 of this License; or
- b) Requiring preservation of specified reasonable legal notices or author attributions in that material or in the Appropriate Legal Notices displayed by works containing it; or
- c) Prohibiting misrepresentation of the origin of that material, or requiring that modified versions of such material be marked in reasonable ways as different from the original version; or
- d) Limiting the use for publicity purposes of names of licensors or authors of the material; or
- e) Declining to grant rights under trademark law for use of some trade names, trademarks, or service marks; or
- f) Requiring indemnification of licensors and authors of that material by anyone who conveys the material (or modified versions of it) with contractual assumptions of liability to the recipient, for any liability that these contractual assumptions directly impose on those licensors and authors.

All other non-permissive additional terms are considered *further restrictions* within the meaning of section 10. If the Program as you received it, or any part of it, contains a notice stating that it is governed by this License along with a term that is a further restriction, you may remove that term. If a license document contains a further restriction but permits relicensing or conveying under this License, you may add to a covered work material governed by the terms of that license document, provided that the further restriction does not survive such relicensing or conveying.

If you add terms to a covered work in accord with this section, you must place, in the relevant source files, a statement of the additional terms that apply to those files, or a notice indicating where to find the applicable terms.

Additional terms, permissive or non-permissive, may be stated in the form of a separately written license, or stated as exceptions; the above requirements apply either way.

8. Termination.

You may not propagate or modify a covered work except as expressly provided under this License. Any attempt otherwise to propagate or modify it is void, and will automatically terminate your rights under this License (including any patent licenses granted under the third paragraph of section 11).

However, if you cease all violation of this License, then your license from a particular copyright holder is reinstated

- a) provisionally, unless and until the copyright holder explicitly and finally terminates your license, and
- b) permanently, if the copyright holder fails to notify you of the violation by some reasonable means prior to 60 days after the cessation.

Moreover, your license from a particular copyright holder is reinstated permanently if the copyright holder notifies you of the violation by some reasonable means, this is the first time you have received notice of violation of this License (for any work) from that copyright holder, and you cure the violation prior to 30 days after your receipt of the notice.

Termination of your rights under this section does not terminate the licenses of parties who have received copies or rights from you under this License. If your rights have been terminated and not permanently reinstated, you do not qualify to receive new licenses for the same material under section 10.

9. Acceptance Not Required for Having Copies.

You are not required to accept this License in order to receive or run a copy of the Program. Ancillary propagation of a covered work occurring solely as a consequence of using peer-to-peer transmission to receive a copy likewise does not require acceptance. However, nothing other than this License grants you permission to propagate or modify any covered work. These actions infringe copyright if you do not accept this License. Therefore, by modifying or propagating a covered work, you indicate your acceptance of this License to do so.

10. Automatic Licensing of Downstream Recipients.

Each time you convey a covered work, the recipient automatically receives a license from the original licensors, to run, modify and propagate that work, subject to this License. You are not responsible for enforcing compliance by third parties with this License.

An *entity transaction* is a transaction transferring control of an organization, or substantially all assets of one, or subdividing an organization, or merging organizations. If propagation of a covered work results from an entity transaction, each party to that transaction who receives a copy of the work also receives whatever licenses to the work the party's predecessor in interest had or could give under the previous paragraph, plus a right to possession of the Corresponding Source of the work from the predecessor in interest, if the predecessor has it or can get it with reasonable efforts.

You may not impose any further restrictions on the exercise of the rights granted or affirmed under this License. For example, you may not impose a license fee, royalty, or other charge for exercise of rights granted under this License, and you may not initiate litigation (including a cross-claim or counterclaim in a lawsuit)

alleging that any patent claim is infringed by making, using, selling, offering for sale, or importing the Program or any portion of it.

11. Patents.

A *contributor* is a copyright holder who authorizes use under this License of the Program or a work on which the Program is based. The work thus licensed is called the contributor's *contributor version*.

A contributor's *essential patent claims* are all patent claims owned or controlled by the contributor, whether already acquired or hereafter acquired, that would be infringed by some manner, permitted by this License, of making, using, or selling its contributor version, but do not include claims that would be infringed only as a consequence of further modification of the contributor version. For purposes of this definition, *control* includes the right to grant patent sublicenses in a manner consistent with the requirements of this License.

Each contributor grants you a non-exclusive, worldwide, royalty-free patent license under the contributor's essential patent claims, to make, use, sell, offer for sale, import and otherwise run, modify and propagate the contents of its contributor version.

In the following three paragraphs, a *patent license* is any express agreement or commitment, however denominated, not to enforce a patent (such as an express permission to practice a patent or covenant not to sue for patent infringement). To *grant* such a patent license to a party means to make such an agreement or commitment not to enforce a patent against the party.

If you convey a covered work, knowingly relying on a patent license, and the Corresponding Source of the work is not available for anyone to copy, free of charge and under the terms of this License, through a publicly available network server or other readily accessible means, then you must either

1. cause the Corresponding Source to be so available, or
2. arrange to deprive yourself of the benefit of the patent license for this particular work, or
3. arrange, in a manner consistent with the requirements of this License, to extend the patent license to downstream recipients.

Knowingly relying means you have actual knowledge that, but for the patent license, your conveying the covered work in a country, or your recipient's use of the covered work in a country, would infringe one or more identifiable patents in that country that you have reason to believe are valid.

If, pursuant to or in connection with a single transaction or arrangement, you convey, or propagate by procuring conveyance of, a covered work, and grant a patent license to some of the parties receiving the covered work authorizing them to use, propagate, modify or convey a specific copy of the covered work, then the patent license you grant is automatically extended to all recipients of the covered work and works based on it.

A patent license is *discriminatory* if it does not include within the scope of its coverage, prohibits the exercise of, or is conditioned on the non-exercise of one or more of the rights that are specifically granted under this License. You may not convey a covered work if you are a party to an arrangement with a third party that is in the business of distributing software, under which you make payment to the third party based on the extent of your activity of conveying the work, and under which the third party grants, to any of the parties who would receive the covered work from you, a discriminatory patent license

- a) in connection with copies of the covered work conveyed by you (or copies made from those copies), or
- b) primarily for and in connection with specific products or compilations that contain the covered work, unless you entered into that arrangement, or that patent license was granted, prior to 28 March 2007.

Nothing in this License shall be construed as excluding or limiting any implied license or other defenses to infringement that may otherwise be available to you under applicable patent law.

12. No Surrender of Others' Freedom.

If conditions are imposed on you (whether by court order, agreement or otherwise) that contradict the conditions of this License, they do not excuse you from the conditions of this License. If you cannot convey a covered work so as to satisfy simultaneously your obligations under this License and any other pertinent obligations, then as a consequence you may not convey it at all. For example, if you agree to terms that obligate you to collect a royalty for further conveying from those to whom you convey the Program, the only way you could satisfy both those terms and this License would be to refrain entirely from conveying the Program.

13. Use with the GNU Affero General Public License.

Notwithstanding any other provision of this License, you have permission to link or combine any covered work with a work licensed under version 3 of the GNU Affero General Public License into a single combined work, and to convey the resulting work. The terms of this License will continue to apply to the part which is the covered work, but the special requirements of the GNU Affero General Public License, section 13, concerning interaction through a network will apply to the combination as such.

14. Revised Versions of this License.

The Free Software Foundation may publish revised and/or new versions of the GNU General Public License from time to time. Such new versions will be similar in spirit to the present version, but may differ in detail to address new problems or concerns.

Each version is given a distinguishing version number. If the Program specifies that a certain numbered version of the GNU General Public License *or any later version* applies to it, you have the option of following the terms and conditions either of that numbered version or of any later version published by the Free Software Foundation. If the Program does not specify a version number of the GNU General Public License, you may choose any version ever published by the Free Software Foundation.

If the Program specifies that a proxy can decide which future versions of the GNU General Public License can be used, that proxy's public statement of acceptance of a version permanently authorizes you to choose that version for the Program.

Later license versions may give you additional or different permissions. However, no additional obligations are imposed on any author or copyright holder as a result of your choosing to follow a later version.

15. Disclaimer of Warranty.

THERE IS NO WARRANTY FOR THE PROGRAM, TO THE EXTENT PERMITTED BY APPLICABLE LAW. EXCEPT WHEN OTHERWISE STATED IN WRITING THE COPYRIGHT HOLDERS AND/OR OTHER PARTIES PROVIDE THE PROGRAM *AS IS* WITHOUT WARRANTY OF ANY KIND, EITHER EXPRESSED OR IMPLIED, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE. THE ENTIRE RISK AS TO THE QUALITY AND PERFORMANCE OF THE PROGRAM IS WITH YOU. SHOULD THE PROGRAM PROVE DEFECTIVE, YOU ASSUME THE COST OF ALL NECESSARY SERVICING, REPAIR OR CORRECTION.

16. Limitation of Liability.

IN NO EVENT UNLESS REQUIRED BY APPLICABLE LAW OR AGREED TO IN WRITING WILL ANY COPYRIGHT HOLDER, OR ANY OTHER PARTY WHO MODIFIES AND/OR CONVEYS THE PROGRAM AS PERMITTED ABOVE, BE LIABLE TO YOU FOR DAMAGES, INCLUDING ANY GENERAL, SPECIAL, INCIDENTAL OR CONSEQUENTIAL

DAMAGES ARISING OUT OF THE USE OR INABILITY TO USE THE PROGRAM (INCLUDING BUT NOT LIMITED TO LOSS OF DATA OR DATA BEING RENDERED INACCURATE OR LOSSES SUSTAINED BY YOU OR THIRD PARTIES OR A FAILURE OF THE PROGRAM TO OPERATE WITH ANY OTHER PROGRAMS), EVEN IF SUCH HOLDER OR OTHER PARTY HAS BEEN ADVISED OF THE POSSIBILITY OF SUCH DAMAGES.

17. Interpretation of Sections 15 and 16.

If the disclaimer of warranty and limitation of liability provided above cannot be given local legal effect according to their terms, reviewing courts shall apply local law that most closely approximates an absolute waiver of all civil liability in connection with the Program, unless a warranty or assumption of liability accompanies a copy of the Program in return for a fee.

END OF TERMS AND CONDITIONS

How to Apply These Terms to Your New Programs

If you develop a new program, and you want it to be of the greatest possible use to the public, the best way to achieve this is to make it free software which everyone can redistribute and change under these terms.

To do so, attach the following notices to the program. It is safest to attach them to the start of each source file to most effectively state the exclusion of warranty; and each file should have at least the *copyright* line and a pointer to where the full notice is found.

```
<one line to give the program's name and a brief idea of what it does.>
Copyright (C) <year>  <name of author>

This program is free software: you can redistribute it and/or modify
it under the terms of the GNU General Public License as published by
the Free Software Foundation, either version 3 of the License, or
(at your option) any later version.

This program is distributed in the hope that it will be useful,
but WITHOUT ANY WARRANTY; without even the implied warranty of
MERCHANTABILITY or FITNESS FOR A PARTICULAR PURPOSE.  See the
GNU General Public License for more details.

You should have received a copy of the GNU General Public License
along with this program.  If not, see <https://www.gnu.org/licenses/>.
```

Also add information on how to contact you by electronic and paper mail.

If the program does terminal interaction, make it output a short notice like this when it starts in an interactive mode:

```
<program>  Copyright (C) <year>  <name of author>
This program comes with ABSOLUTELY NO WARRANTY; for details type `show w'.
This is free software, and you are welcome to redistribute it
under certain conditions; type `show c' for details.
```

The hypothetical commands `show w` and `show c` should show the appropriate parts of the General Public License. Of course, your program's commands might be different; for a GUI interface, you would use an *about box*.

You should also get your employer (if you work as a programmer) or school, if any, to sign a *copyright disclaimer* for the program, if necessary. For more information on this, and how to apply and follow the GNU GPL, see https://www.gnu.org/licenses/.

The GNU General Public License does not permit incorporating your program into proprietary programs. If your program is a subroutine library, you may consider it more useful to permit linking proprietary applications with the library. If this is what you want to do, use the GNU Lesser General Public License instead of this License. But first, please read https://www.gnu.org/philosophy/why-not-lgpl.html.