Access Server

User's and Developer's Guide

Bluegiga Technologies

Access Server: User's and Developer's Guide
by Bluegiga Technologies

Published 2007-01-22 (3.1)

Table of Contents

List of Tables

Chapter 1. Introduction to Access Server

Bluegiga's WRAP™ product family offers for device manufacturers, integrators, companies and developers a simple and fast way to set-up wireless communication systems between standard or proprietary devices, networks, machines and instruments.

Access Server is a cutting edge wireless Bluetooth router. It supports multiple communication standards including Ethernet, WiFi, and GSM/GPRS enabling full media-independent TCP/IP connectivity. Access Server is easy to deploy and manage in existing wired and wireless networks without compromising speed or security. For rapid deployment, Access Server configurations can easily be copied from one device to another by using USB memory dongles. The device can be conveniently managed and upgraded remotely over SSH secured links. By using Simple Network Management Protocol (SNMP), Access Servers can also be connected to the customer's management and monitoring systems.

Access Server usage scenarios and applications:

- Point-of-sales systems
- Logistics and transportation systems
- Telemetry and machine-to-machine systems
- Medical and healthcare systems
- Fitness and sport telemetry systems
- Cable replacement
- Content and application distribution to mobile phones and PDAs

Access Server key features:

- Enables Bluetooth networking between multiple devices and networks
- Serves up to 21 simultaneous Bluetooth connections
- Offers an open platform for adding local applications
- Acts as a transparent router or bridge
- Supports all key communication medias:
  - Bluetooth
  - Ethernet
  - WiFi, GSM and GPRS with a Compact Flash card
  - USB and RS232
- Incorporates a packet filtering firewall
- Is fast and easy to install
- Supports all relevant Bluetooth profiles and APIs
- 100 meter range / Software configurable to support 10 meter range
- DHCP support for plug-and-play installation
- Uncompromised security: SSH, firewall, and 128 bit Bluetooth encryption

- Simple and secure mounting accessory available
- Bluetooth, CE, and FCC certified
- Compliant with Bluetooth 1.1, 1.2 and 2.0 Specification

1.1. Licenses and Warranty

Warning

Bluegiga Technologies is hereby willing to license the enclosed WRAP product and its documentation under the condition that the terms and conditions described in the License Agreement are understood and accepted. The License Agreement is supplied within every WRAP product both in hard copy. It is also available on-line at http://bluegiga.com/as/current/doc/eula.pdf. The use of the WRAP product will indicate your assent to the terms. If you do not agree to these terms, Bluegiga Technologies will not license the software and documentation to you, in which event you should return this complete package with all original materials, equipment, and media.

Some software components are licensed under the terms and conditions of an open source license. Details can be found in Appendix C. Upon request, Bluegiga will distribute a complete machine-readable copy of the source of the aforementioned open source software components during a period of three (3) years from the release date of the software. Delivery costs of the source code will be charged from the party requesting the source code.

The Bluegiga WRAP Product Limited Warranty Statement is available on-line at http://bluegiga.com/as/current/doc/warranty.pdf.

1.2. Bluegiga Technologies Contact Information

Please see http://www.bluegiga.com/ for news and latest product offers. For more information, contact <sales@bluegiga.com>.

Please check http://bluegiga.com/as/ for software and documentation updates.

Please contact <support@bluegiga.com> if you need more technical support. To speed up the processing of your support request, please include as detailed information on your product and your problem situation as possible.

Please begin your email with the following details:

- Access Server product type
- Access Server product serial number
- Access Server software version
- End customer name
- Date of purchase

Chapter 2. Getting Started with Access Server

Access Server can be controlled in three ways:

- by using the WWW interface
- by entering commands and using applications at the Access Server shell prompt
- by sending and/or retrieving files to/from Access Server.

Note: The default username is **`root`** and the default password is **`buffy`**.

2.1. Powering Up

To get started with Access Server, connect it to your local area network (LAN) by using an Ethernet cable, and connect the power adapter. Access Server will power up and retrieve the network settings from your network's DHCP server.

Access Server will also use Zeroconf (also known as Zero Configuration Networking or Automatic Private IP Addressing) to get an unique IP address in the 169.254.x.x network. Most operating systems also support this. In other words, you can connect your controlling laptop with a cross-over Ethernet cable to Access Server, then power up Access Server, and the devices will automatically have unique IP addresses in the 169.254.x.x network.

Note: If you need to configure the network settings manually and cannot connect first by using Zeroconf, you can do it by using the management console. For more information, see Section 2.3.1.

The physical interface locations of Access Server are described in Figure 2-1 and Figure 2-2.

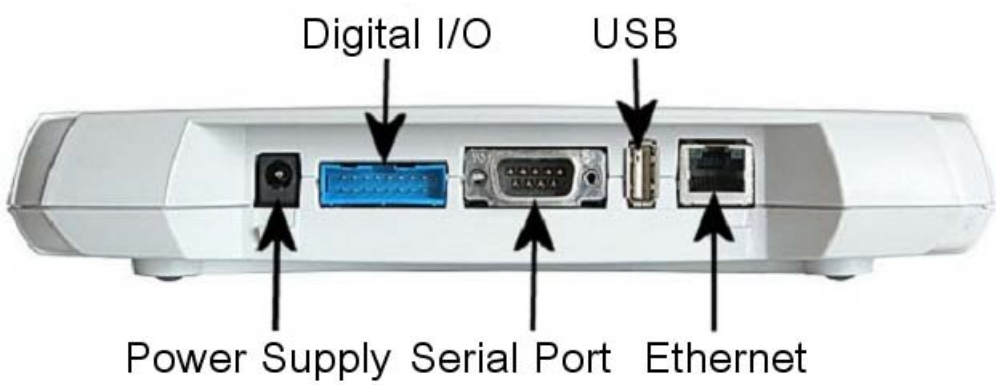

Figure 2-1. Access Server Connectors

Note: There is no power switch in Access Server. The adapter is the disconnection device; the socket-outlet shall be installed near the equipment and shall be easily accessible. Unplug and plug the power adapter to switch the power on and off. The power led in Figure 2-2 is on when the power adapter is connected.

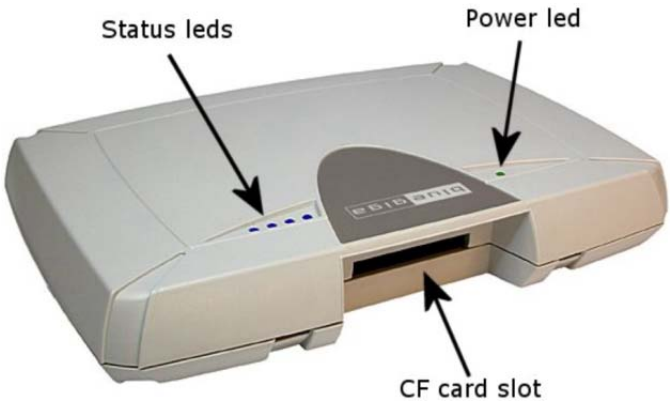

Figure 2-2. Access Server LEDs

All the blue status LEDs are turned off when the boot procedure is finished and Access Server is ready to be connected.

2.2. WWW Interface

Most Access Server functionality can be controlled through the WWW interface by using any standard WWW browser.

The wrapfinder application (see Figure 2-3), available for the Windows operating system from Bluegiga Techforum (http://www.bluegiga.com/techforum/) provides an easy-to-use interface for finding Access Servers (with SW version 2.1.0 or later) in the local area network.

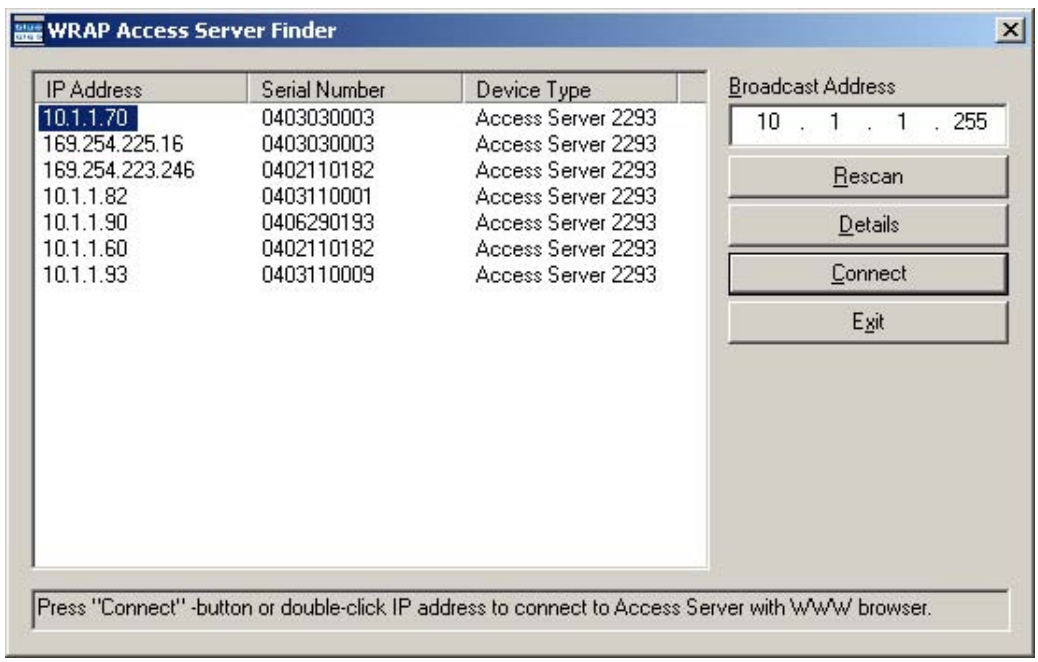

Figure 2-3. Access Server Finder Application

The wrapfinder automatically identifies the broadcast address of the network it runs in, and shows the IP addresses, serial numbers, and Access Server device types it could find by using

UDP broadcast when it was launched.

> **Note:** Normally, there are two entries for each Access Server. Use the one with the IP address in your local area network. Use the one with the 169.254.x.x, the Zeroconf network address, when it is the only one shown.

You can change the broadcast address used for finding Access Servers. A new scan can be done by clicking Rescan.

Select an Access Server by clicking its IP address, and click Details to see more information (such as the Bluetooth addresses and friendly names) on Access Server. See Figure 2-4 for details.

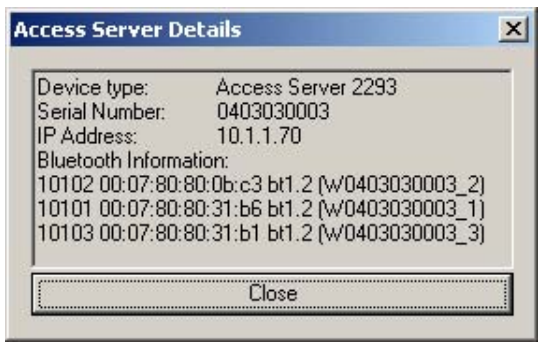

Figure 2-4. Details Dialog of Access Server Finder

Click Connect or double-click an IP address to connect to the selected Access Server by using a WWW browser.

Click Exit to close the program.

> **Note:** To find Access Server's IP address without wrapfinder, see Section 2.3.2.

To access the WWW interface, enter the IP address of Access Server to the browser's address field and press **Enter** (see Figure 2-5).

Figure 2-5. Access Server WWW Interface

From the top-level page, click Setup to log in to the configuration interface. The default username is **`root`** and the default password is **`buffy`** (see Figure 2-6).

Figure 2-6. WWW Login Prompt for Access Server Setup

After logging in, you can configure several Access Server settings (see Figure 2-7). These are discussed in detail in Section 2.4.

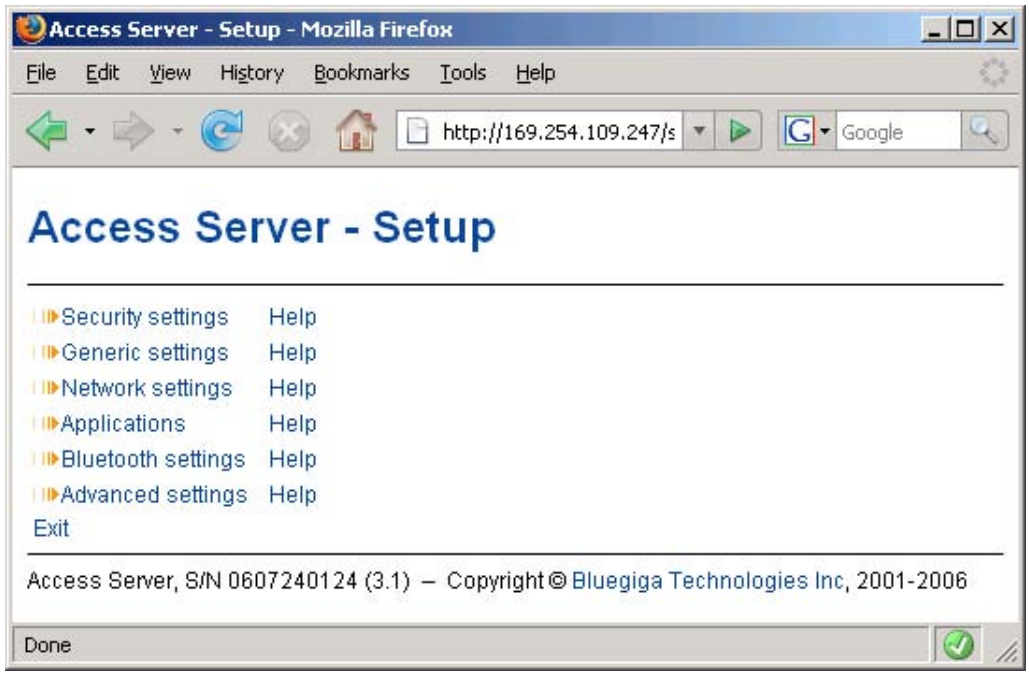

Figure 2-7. The WWW Configuration Interface of Access Server

2.3. Shell Prompt Access

Shell prompt access may be needed for advanced controlling operations that cannot be performed by using the WWW interface.

You can get to the shell prompt by using either SSH or the management console. The management console is only needed to change the network configuration settings if you cannot configure the network by using DHCP or Zeroconf. The management console is connected to Access Server with a serial cable. All further controlling activities can be performed remotely using SSH sessions over Ethernet or Bluetooth LAN/PAN connection.

If you can establish an SSH connection from a device that has Bluetooth LAN Access or PAN profile support, you do not need the management console. Just connect to Access Server by using LAN Access or PAN profile. Access Server can be seen in Bluetooth inquiries as "Wserialno_n", where "serialno" is the serial number of the device and "n" is the number of the Bluetooth baseband in question (model 2293 has three Bluetooth basebands, any of which can be connected). After you have connected to the server (no PIN code, username or password needed), establish an SSH connection to the device at the other end of the connection, typically 192.168.160.1. You can also use the wrapfinder application to find the IP address (see Section 2.2 for details).

Note: Bluetooth LAN Access and PAN profiles are disabled by default. Use the WWW interface to enable them, if needed. The PAN profile can also be enabled by sending the `enable-pan.wpk` file (available on-line at http://bluegiga.com/as/current/enable-pan.wpk) to Access Server by using Bluetooth Object Push profile or by inserting a USB memory dongle with the file in its root directory to Access Server's USB port.
Note: The default username is **`root`** and the default password is **`buffy`**.

2.3.1. Management Console

If you do not have a Bluetooth LAN/PAN client and if Access Server is not connected to your LAN, or if you do not know the IP address given to Access Server, you can get the first shell prompt access by using the management console.

To setup the management console, proceed as follows:

1. Have a PC with a free COM port.
2. Power off Access Server.
3. Configure your terminal application, such as HyperTerminal in Windows, to use the settings below for your computer's free COM port

Setting	Value
Speed	115200bps
Data Bits	8
Parity	None
Stop Bits	1
Flow Control	None

Table 2-1. The Management Console Port Settings

4. Connect the serial cable shipped with Access Server to your PC's free COM port.
5. Connect the serial cable to the management (user) port in Access Server (see Figure 2-1).
6. Power on Access Server.
7. Enter letter **b** in the terminal application during the first five seconds, while the blue LEDs in Access Server turn on one by one.
8. The management console is now activated and you can see the boot log in your terminal window.

   Note: The boot process may stop at the following U-Boot prompt:

   ```
   Hit any key to stop autoboot:  0
   U-Boot>
   ```

   If this happens, enter command **`boot`** to continue to boot Linux.
9. Wait for the device to boot up and end with the following prompt:

   ```
   Please press Enter to activate this console.
   ```

10. Press **Enter** to activate the console. You will be logged in as **`root`** in directory `/root`:

    ```
    [root@wrap root]
    ```

11. You can now control Access Server from the management console.

2.3.2. Accessing Remotely

When Access Server is connected to a LAN, it tries to get the IP address by using DHCP and Zeroconf by default. You can then use the wrapfinder application to find the IP address (see

Section 2.2).

If you cannot get the IP address by using the wrapfinder, another way to see the IP address of Access Server is to connect with a management console (see previous section), power on the unit and, after the system is up and running, give the **ifconfig nap** command. The `inet addr` field for the `nap` interface contains the IP address of Access Server. For example, in the following capture from the management console, the IP address is `192.168.42.3`.

```
[root@wrap /]$ ifconfig nap
nap       Link encap:Ethernet  HWaddr 00:07:80:00:BF:01
          inet addr:192.168.42.3  Bcast:192.168.42.255  Mask:255.255.255.0
          inet6 addr: fe80::207:80ff:fe00:bf01/64 Scope:Link
          UP BROADCAST MULTICAST  MTU:1500  Metric:1
          RX packets:12635 errors:0 dropped:0 overruns:0 frame:0
          TX packets:8 errors:0 dropped:0 overruns:0 carrier:0
          collisions:0 txqueuelen:100
          RX bytes:1686246 (1.6 MiB)  TX bytes:1640 (1.6 KiB)
          Interrupt:24 Base address:0xc000
```

You can use this address to connect to Access Server remotely over SSH, SCP or SFTP.

Note: The default username is **root** and the default password is **buffy**.

2.3.3. Transferring Files to/from Access Server

You can transfer files to and from Access Server by using, for example:

- SCP (secure copy over SSH)
- SFTP (secure FTP connection over SSH)
- FTP (plain FTP connection)
  Note: FTP is disabled by default for security reasons. Use SFTP instead.
  Tip: If enabled, use the integrated FTP client on the Internet Explorer (type ftp://root:buffy@wrap-ip-address/ in the address bar)
- Bluetooth OBEX (Object Push and File Transfer Profiles) to/from directory `/tmp/obex` in Access Server
- NFS (mount an NFS share from a remote device as a part of Access Server's file system)
- SSHFS (mount an Access Server directory over SSH as a part of any other Linux host file system)

  To download and install SSHFS, visit http://fuse.sourceforge.net/sshfs.html.
- USB memory dongle (see Section 3.4 for more information).
- Xmodem/Ymodem/Zmodem (use **rz/rx/rb/sz/sx/sb** commands from the management console)

For examples of transferring files, see Section 6.3.4.

2.4. Introduction to Configuration

When Access Server is installed and powered up for the first time, the default configuration settings are being used. With these settings, Access Server automatically configures its network settings assuming that it is connected to a LAN network with a DHCP server running. Additionally, Access Server also uses Zero Configuration Networking (also known as Automatic Private IP Addressing) to connect to the 169.254.x.x network, which can be used if the network has no DHCP server.

After booting up, the only Bluetooth profiles enabled are the Object Push and File Transfer Profiles, used to send files to/from Access Server.

More Bluetooth profiles can be enabled, and most of Access Server settings can be configured by using the **setup** application. It has a WWW interface at http://wrap-ip/setup but it can also be run at the command line.

All configurable settings in the **setup** application are listed in Appendix B with short help texts.

Note: The default username is **`root`** and the default password is **`buffy`**.

2.5. Using the Setup WWW Interface

The easiest way to change Access Server settings is to use the WWW interface. Accessing the WWW interface is instructed in Section 2.2.

A typical WWW configuration page is shown in Figure 2-8 (This page can be found at Setup ⟶ Security settings)

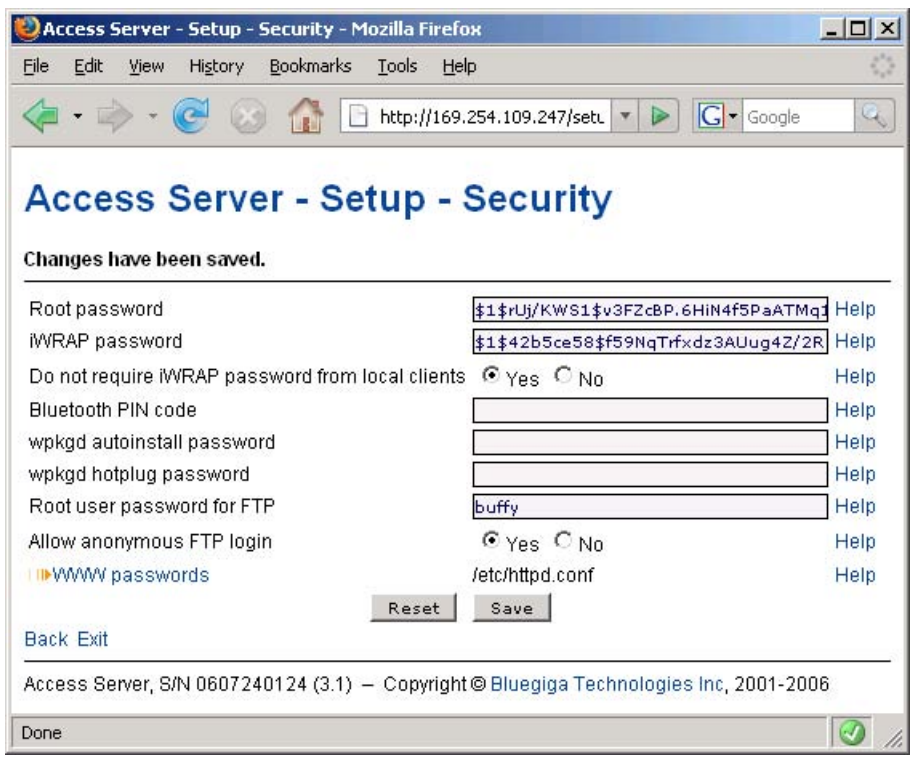

Figure 2-8. Example WWW Setup Page

The different parts of the WWW Setup page are discussed in the following list:

- Status area

  The status area serves two purposes:

  - It indicates that the changes are permanently saved when the user clicks the Save button (or when the user clicks a toggling Yes/No link).
  - If invalid values were entered in one or more fields, an error message is shown in this area (see Figure 2-9).

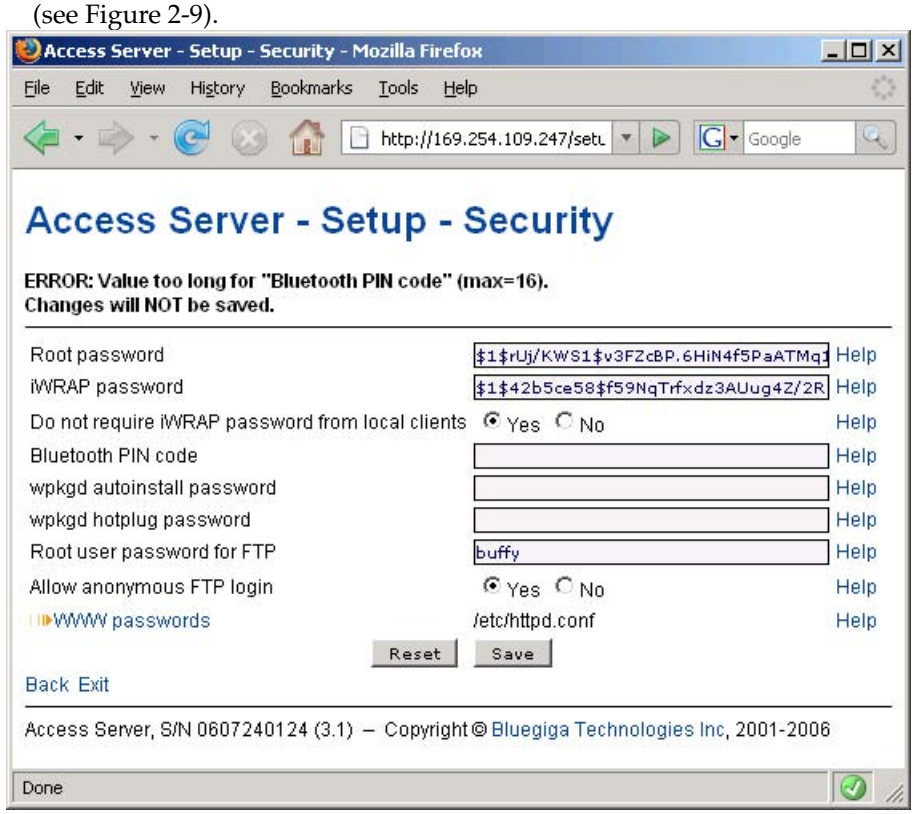

Figure 2-9. Trying to Save an Invalid Input

Note: It is typically necessary to reboot Access Server for the changes to take effect. This can be done through the WWW interface (Advanced settings menu).

- Number or text entry fields

  Most of the configurable settings are text (or number) entry fields. For some fields, such as the IP address or netmask, there are restrictions on the input format. Setup validates the input at save time and accepts valid data only. The fields with errors are shown to the user so that mistakes can be fixed (see Figure 2-9).

- Help -link

  Click the Help link to retrieve the setup page again with requested help information displayed. For an example, see Figure 2-10.

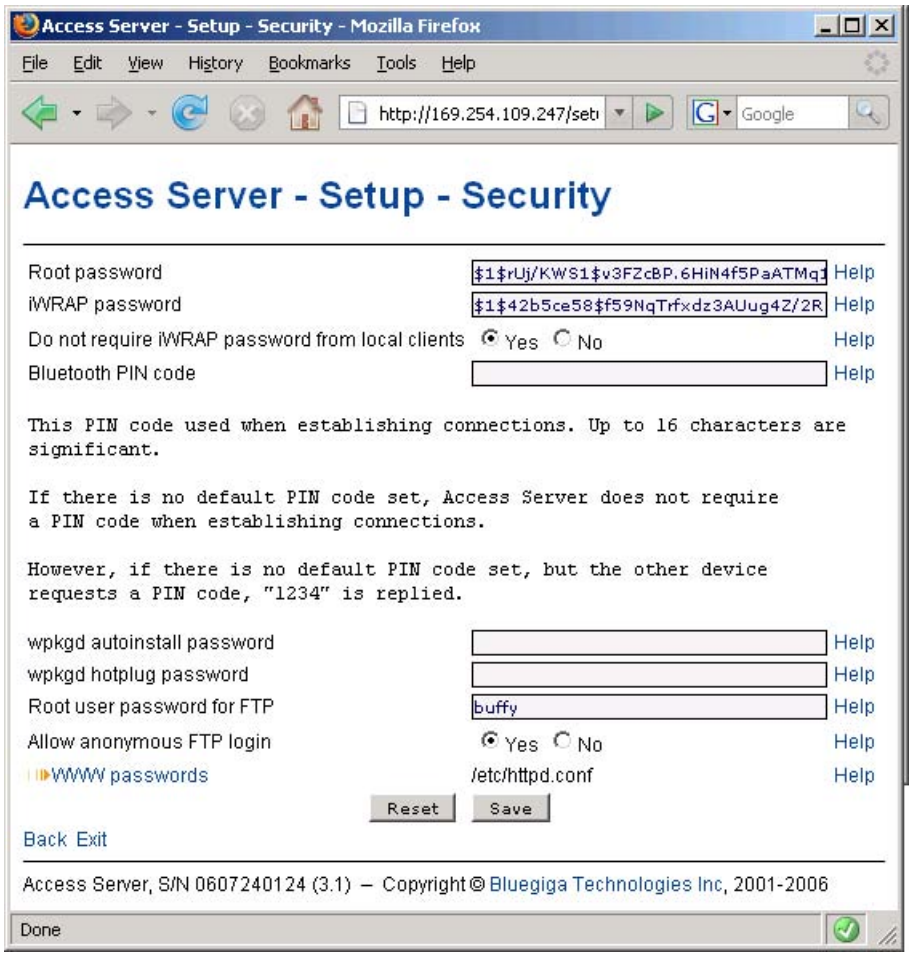

Figure 2-10. Help Links in WWW Setup

Warning

If you have made changes to the settings on the page before clicking Help and not saved them yet, they are lost.

- Yes and No radio buttons

  These buttons are typically used to configure a setting that can be either enabled or disabled, and this setting has no effect on the visibility of other settings.

- Link to a configuration file

  Some of the configurable settings are actually editable configuration files, such as `/etc/httpd.conf` for WWW passwords. Clicking the link will retrieve the file for editing in the browser window, or create a new file, if it does not exist. See Figure 2-11.

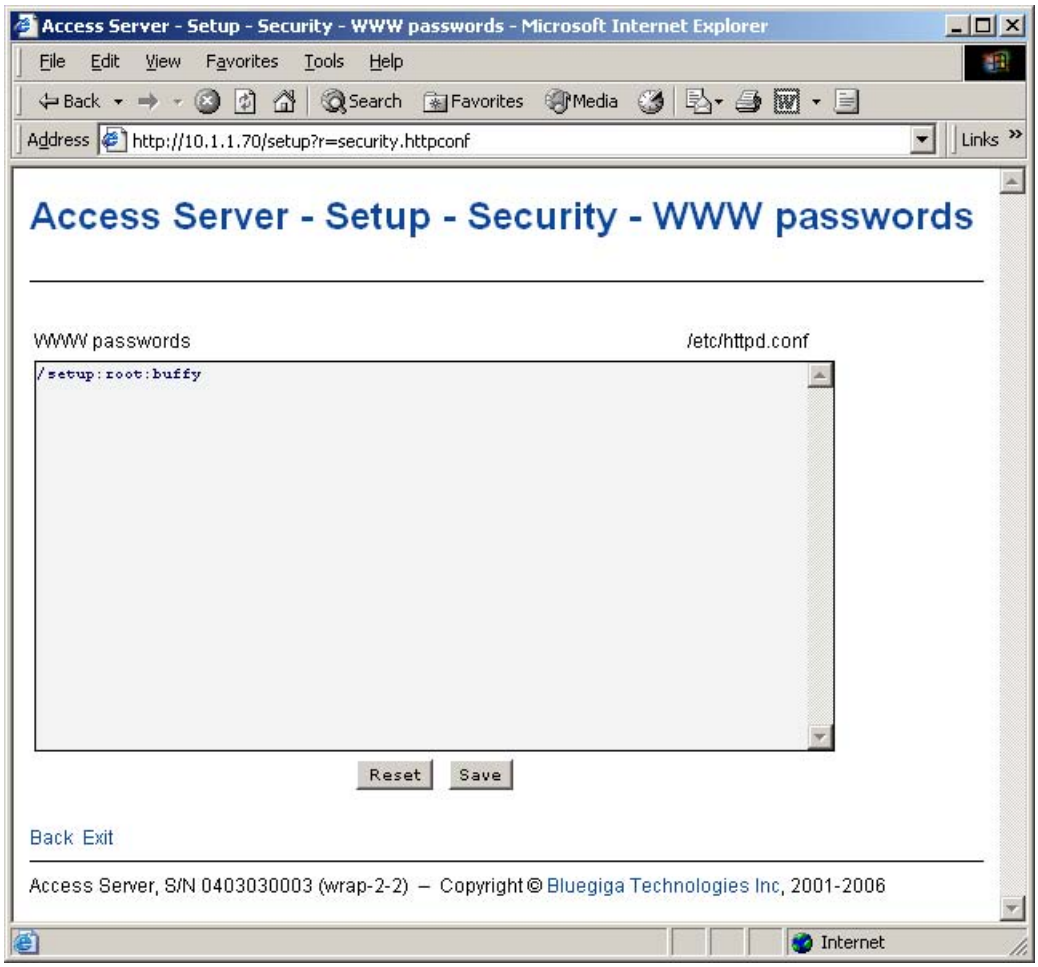

Figure 2-11. Editing Files in WWW Setup

Note: You can edit any file through the WWW Setup. to edit files, navigate to Setup ⟶ Advanced setting ⟶ Edit other configuration files.

- Reset button

  Reset button resets the fields to the values currently in use at Access Server. In other words, the Reset button discards unsaved changes.

  Note: The Reset button does *not* make a "factory reset".

- Save button

  Save button sends the WWW page to the setup application for validation. If the values in the fields are valid, they are permanently saved and the page is refreshed with the Changes have been saved. message at the top. The accepted values are shown in the page fields.

  If there were errors in the fields, these are shown as in Figure 2-9.

  Note: It is typically necessary to reboot Access Server for the changes to take effect. This can be done through the WWW interface (Advanced settings menu).

- Back link

  Press the Back link to return to the previous level of the Setup menu hierarchy.

  Note: Pressing the Back link does *not* save changes in the fields on the current page.

- Exit link

  Exit link quits the setup application and returns to the Access Server's main WWW page.

  Note: Pressing the Exit link does *not* save changes in the fields on the current page.

- Toggling Yes/No and on/off links

  Clicking the Yes/No link (see Figure 2-12) immediately changes the setting and saves the change. Typically these links are used display or hide further settings.

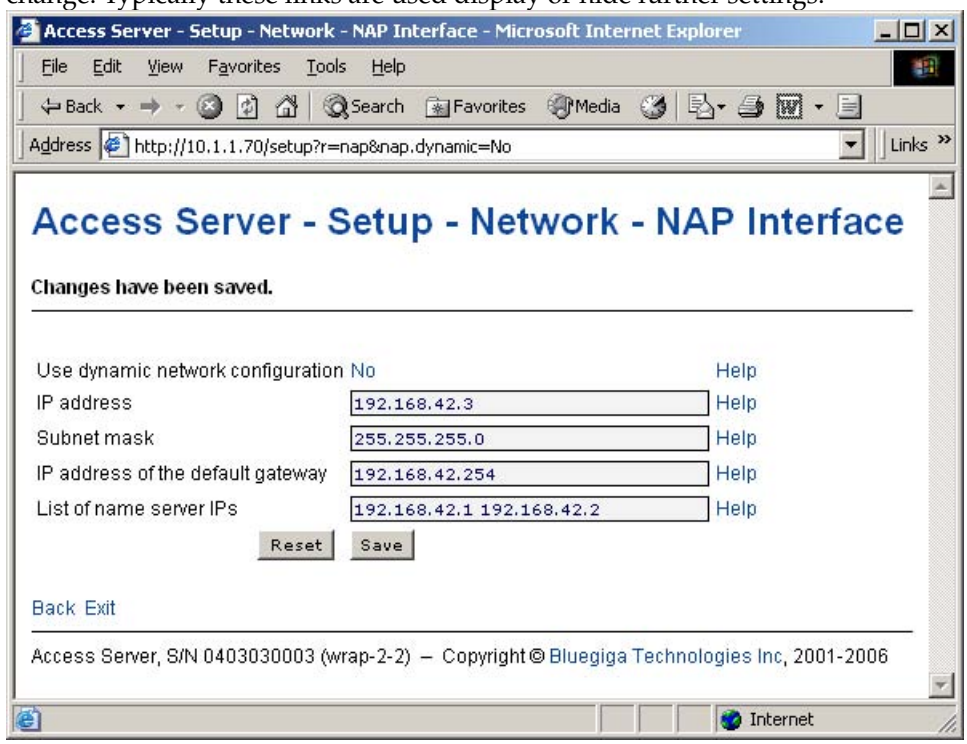

Figure 2-12. Yes / No links in WWW Setup

The on/off links in Setup ⟶ Applications ⟶ Default bootup applications behave in a same way, making and saving the change immediately (see Figure 2-13).

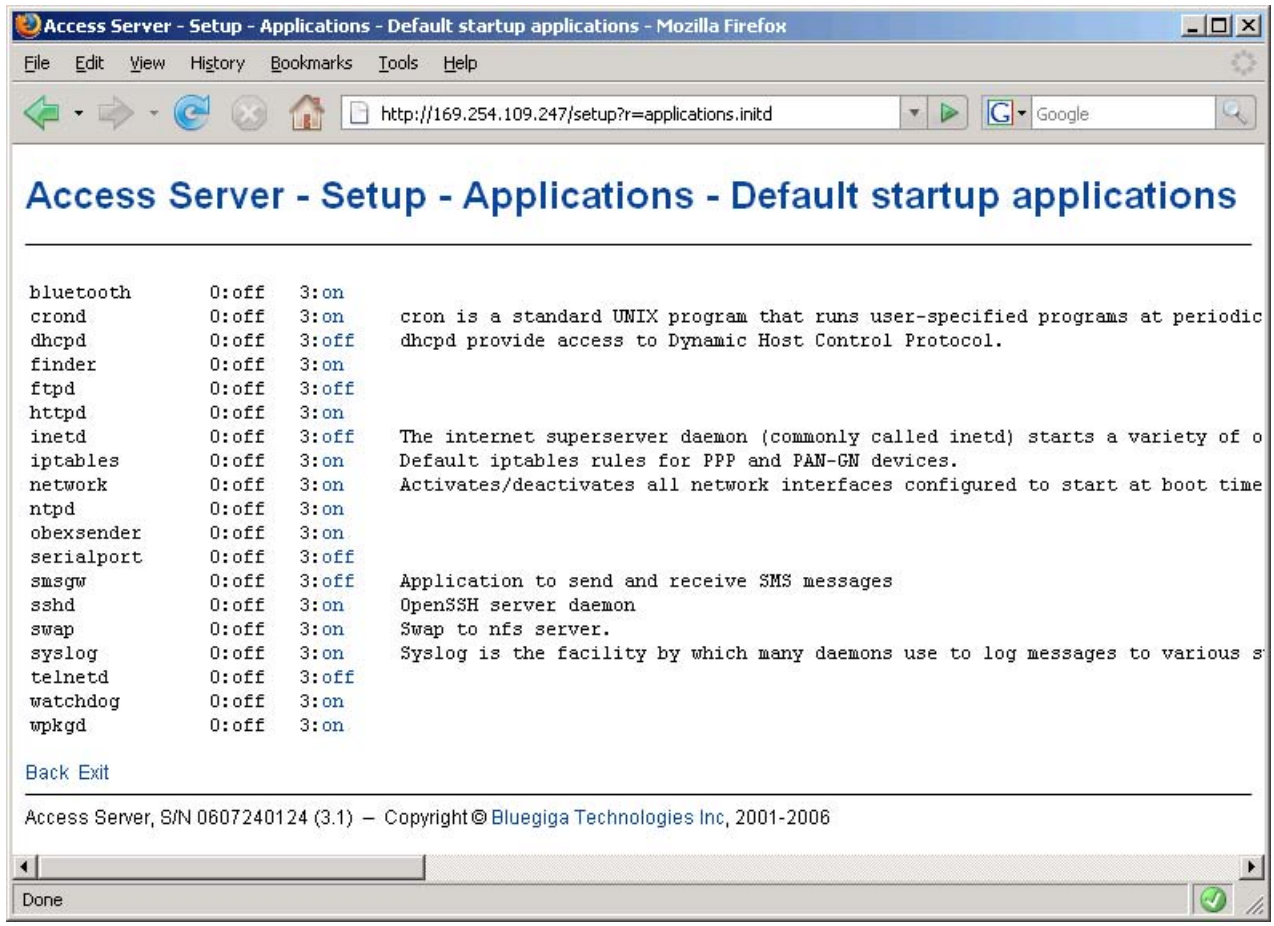

Figure 2-13. Selecting Default Bootup Applications in WWW Setup

Note: To configure the default bootup applications from the command line, use the **chkconfig** command.

- Upload links

The WWW Setup has settings that allow user to upload files to Access Server, for example Setup ⟶ Advanced ⟶ Upload a software update (see Figure 2-14).

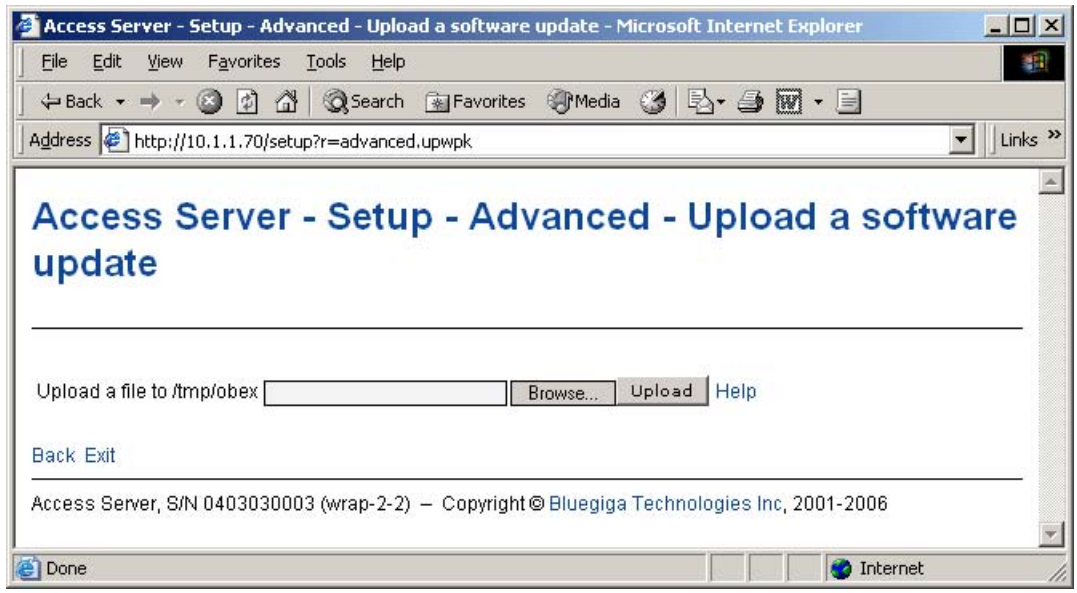

Figure 2-14. Uploading files via WWW Setup

Use the Browse... button to select the file to be uploaded, and send it to Access Server by clicking Upload.

- Browsing files

  Some WWW Setup pages allow users to browse the Access Server file system or part of it, such as Setup ⟶ Advanced ⟶ Browse files (see Figure 2-15).

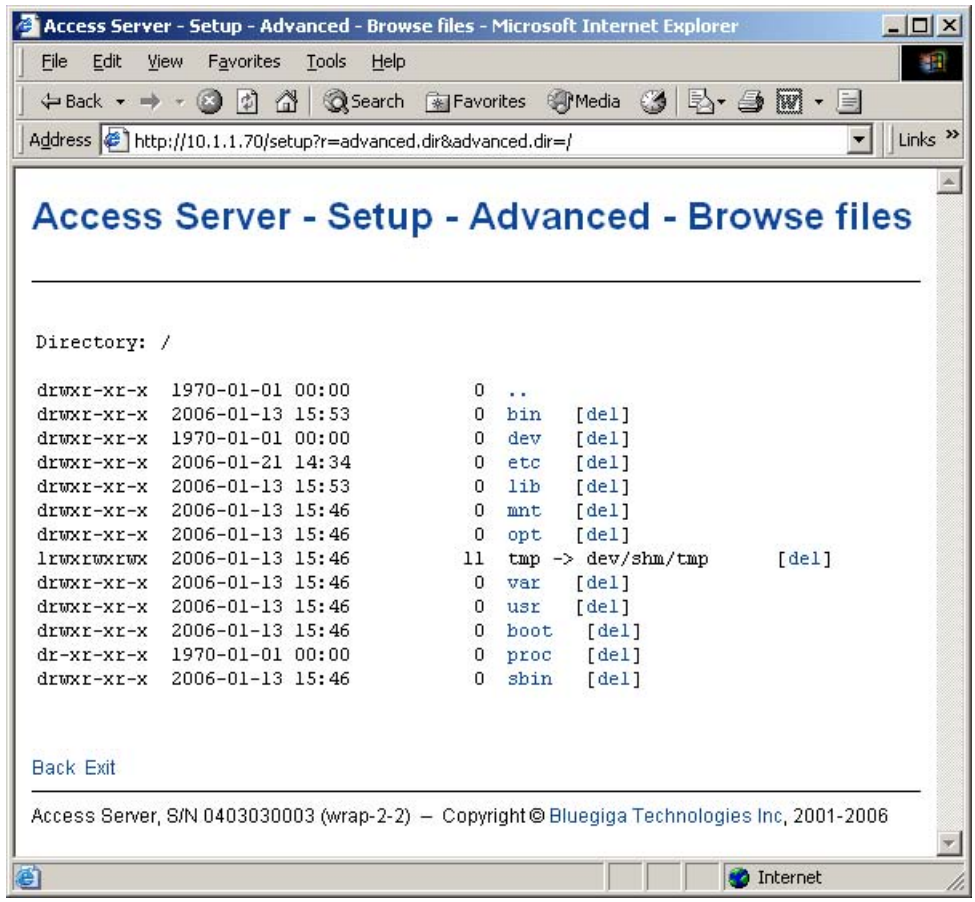

Figure 2-15. Browsing files via WWW Setup

Click the directory names to navigate in the file system.

Click del to delete a file or an empty directory.

Warning

Deletion is not confirmed.

The WWW Setup also has menu items that run commands in Access Server, and show the output in the browser window. Some commands, such as rebooting Access Server, are confirmed before execution.

2.6. Using the setup Command Line Application

The basic configuration settings can also be changed by using the **setup** application at the command line interface.

The **setup** application displays the settings in a hierarchical menu (see Figure 2-16). Navigating the menu is accomplished by entering the number or letter corresponding to the setting to be viewed and/or changed and pressing **Enter**. Pressing only **Enter** either accepts the previous value of the setting or returns to the previous level in the menu hierarchy.

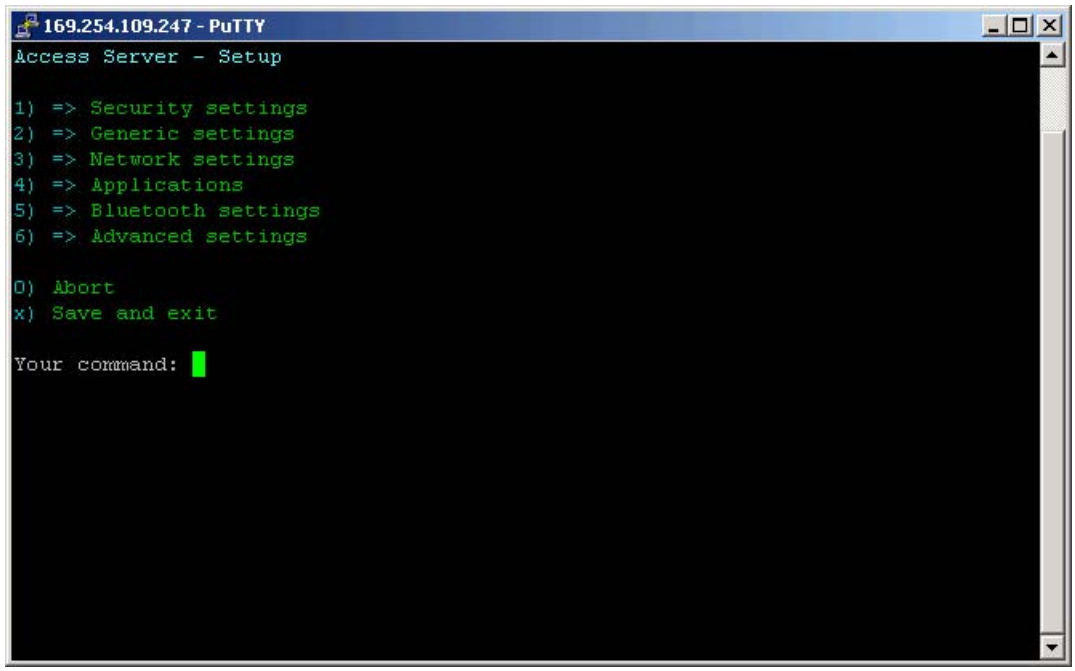

Figure 2-16. Using the setup Command Line Application

Note: Ensure that your terminal application does not send line ends with line feeds. If your terminal sends both CR and LF when you press **Enter**, you cannot navigate in the **setup** application.

2.7. Resetting a Configuration

You can reset the default configuration with the **setup -r** command. The command requires rebooting of Access Server. When the system starts up, the default configuration settings are restored. If you have only changed the configuration by using the **setup** application, the following commands at the Access Server's command prompt will suffice:

```
[root@wrap /]$ setup -r
[root@wrap /]$ reboot
```

Note: This does not reset the edited files to factory defaults; it only affects only the settings changed through the WWW Setup or the **setup** command line application.

2.8. Exporting and Importing Configurations

You can export configuration settings (expect for passwords and the list of default bootup applications) with the following command:

```
[root@wrap /root]$ setup -o > settings.txt
```

The saved settings can later be restored with the following commands:

```
[root@wrap /root]$ setup -m settings.txt
[root@wrap /root]$ reboot
```

Chapter 3. Using the System

This chapter describes the basic features of a Bluegiga Access Server. This includes information on using Access Server as a Bluetooth LAN/PAN Access Point or a Bluetooth Serial Port Cable Replacer, using the Web Server, ObexSender, and WRAP Package Management System. The various ways of uploading content for browsing and/or downloading are also included, as well as getting familiar with the utility applications.

Using the features described in this chapter does not require Access Server Software Development Environment to be installed.

Note: The default username is **`root`** and the default password is **`buffy`**.
Note: Most of the configuration files are in Linux text file format, where the lines end with a single Line Feed (LF, "\n") character. Some applications will not work if the configuration file format is changed to MS-DOS format (this happens, for example, if you transfer the files to Windows for editing with Notepad), where the lines end with both Carriage Return and Line Feed (CR+LF, "\r\n") characters.

3.1. Network Interfaces

The Access Server network interfaces are described in Table 3-1.

Interface	Description
nap	Dynamic virtual Ethernet ("cable") device. This is the device having an IP address. All the programs should use this device instead of eth0.
eth0	Real Ethernet device, which is dynamically linked to the nap device. Do not use this device, use nap instead.
wlan0	Wi-Fi device. In the client mode (default), this device has its own IP address. In the access point mode, it is dynamically linked to the nap device (the default interface).
wifi0	Virtual control device for wlan0. Do not use this device.
gn	Virtual device for PAN-GN connections.
bnep#	These devices are used for incoming and outgoing PAN connections. These devices are created, deleted and linked (to nap or gn) dynamically.
ppp#	These devices are used for incoming and outgoing LAP connections. These devices are created and deleted dynamically. By default, data coming from ppp# is masqueraded to the nap device.

Table 3-1. Access Server Network Interfaces

3.2. Bluetooth

The iWRAP servers (one server in Access Server 2291, three in Access Server 2293) are automatically started at power-up. By default, the Object Push and File Transfer Profiles are activated. The iWRAP servers can be accessed and controlled (by applications or even interactively with a telnet client) through the iWRAP interface, described in Chapter 7. Currently, there can be up to 14 simultaneous Bluetooth connections between a single master iWRAP server and up to seven simultaneous slaves.

3.2.1. iWRAP Password Protection

The access to iWRAP can be password protected. The default password is **buffy**, but it can be set off or changed with the **setup** application (see Section 2.4). The password is case sensitive. The password must be typed in as the first command after the server has replied with "READY."

3.2.2. LAN Access Profile

This profile is not automatically started at boot. The default settings can be changed with the **setup** application (see section Section 2.4), or runtime with the iWRAP interface (see Chapter 7).

Access Server can also act as a LAN Access Client, but in this case it must be controlled manually using iWRAP commands, as described in Chapter 7.

Note: Since Bluetooth specification 1.2, LAN Access Profile has been deprecated.

3.2.3. Serial Port Profile

This profile is not automatically started at boot. The default settings can be changed with the **setup** application (see section Section 2.4).

The Serial Port Profile is used to replace an RS-232 serial cable between two devices with a Bluetooth connection. The physical setup is shown in Figure 3-1.

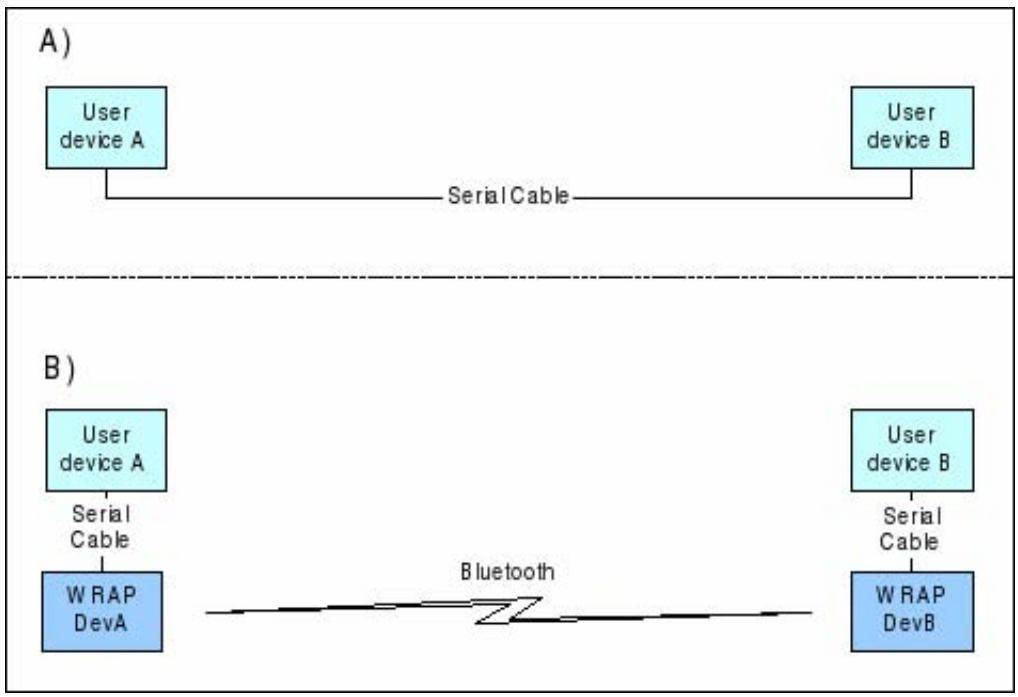

Figure 3-1. Serial Cable Replacement Physical Setup

State A) in the figure is the starting situation with a serial cable connecting the devices. This cable is to be replaced with a Bluetooth connection.

In state B) the long serial connection is replaced with a Bluetooth Serial Port Profile connection between the two Access Server devices. These Access Server devices are then locally connected

to the user devices with (short) serial cables. The cable between user device A and Access Server device A must be a cross-over cable. The cable between user device B and Access Server device B must be similar (direct or cross-over) to the one used in state A).

If RTS/CTS handshaking is used to ensure correct data transfer, the serial cables must have these pins connected. Notice that this handshaking is "local": it takes place between the user device and Access Server. No handshaking between user device A and user device B on the other end of the Bluetooth connection is provided.

If RTS/CTS handshaking is not used, CTS must be connected to DTR.

DCD, DTR, and DSR signals are not supported. This also means that user devices A and B will not be able to tell whether or not the Bluetooth connection is up.

When the physical setup is ready, you can create the Bluetooth connection. By default, the Serial Port Profile is started up at boot with the default settings. That is, listening in DevB mode, at 115200 bps, 8 data bits, no parity, 1 stop bit, and RTS/CTS enabled. To change these settings, use the **setup** application or the WWW Setup interface, as described in Section 2.4.

> **Note:** To enable Serial Port Profile, navigate to Setup ⟶ Applications ⟶ Default bootup applications in the WWW Setup interface, and switch serialport application to off.
> Enabling can also be done from command prompt with command **chkconfig serialport on**.

3.2.4. Object Push and File Transfer Profile

Access Server has two OBEX profiles: Object Push Profile (ObjP) and File Transfer Profile (FTP). You can use these profiles to transfer files easily between different Access Server devices and other devices supporting ObjP/FTP.

The OBEX profiles are handled by forwarding incoming calls to the **obexserver** program, which handles both profiles. The working directory is `/tmp/obex`, and users have full read and write access to that directory. By default, the default contact card `/etc/default.vcf` is copied to that directory at boot.

In the ObjP mode, **obexserver** will prefix received files with the sender's Bluetooth address and iWRAP port number.

Two simple command line utilities, **obexput** and **obexget**, are provided. They can be used to send and retrieve files to and from another Bluetooth device supporting ObjP/FTP.

Usage:

```
obexput [parameters] bdaddr channel file(s)
```

> **Note:** You can use the friendly name instead of Bluetooth address as the "bdaddr" parameter and keywords "OBJP" and "FTP" as the "channel" parameter for automatic service discovery.

Enter either of these commands without parameters to view a short help text for using the command.

A non-zero return value indicates an error. The reason for this error is printed to the terminal.

> **Tip:** Special parameters and the iWRAP interface (see Chapter 7) **obexput** command can be easily used from the user application as follows:
>
> ```
> CALL bdaddr OBJP FORK \"/usr/bin/obexput - 1 filename\"
> ```

– as bdaddr and `1` as channel tells **obexput** that it will be launched by the iWRAP server, and that data connection is bound to standard input and output.

3.2.5. PAN Profiles

Access Server has support for all PAN profile modes: Personal Area Network User (PANU), Network Access Point (NAP) and Generic Networking (GN). Accepting incoming PAN connections to any of these modes is disabled by default for security reasons.

Access Server can be configured to accept incoming PAN connections and the default settings can be changed by using the **setup** application (see section Section 2.4).

The Network Access Point mode is the most useful PAN profile mode. You can enable it by sending the `enable-pan.wpk` file (available on-line at http://bluegiga.com/as/current/enable-pan.wpk) to Access Server by using the Bluetooth Object Push profile. Alternatively, you can copy the file to the root of a USB memory dongle and insert the dongle to Access Server's USB port.

The device creating the PAN connection decides upon the modes to be used. Access Server automatically handles incoming connections. Access Server can also act as a PAN client, but in this case it must be controlled manually by using the iWRAP interface, described in Chapter 7.

3.2.6. Changing the Bluetooth Range

The transmit power of Access Server is configurable. By default, class 1 (100 meter range) settings are used. The settings can be changed down to "class 2" (10 meter range) settings with the **b2b_class2** command, or even lower with the **b2b_class3** command. Class 1 settings can be restored with the **b2b_class1** command.

After **b2b_class#** is given, it is recommended to reboot Access Server once to restart ObexSender and other applications connected to the iWRAP server(s).

> **Note:** If the operation is successful, you get one `Can't open baseband` message with Access Server model 2293 and three messages with the 2291 model.

3.2.7. BTCLI - iWRAP Command Line Interface Utility

You can send commands to an iWRAP server by using the **btcli** application.

Usage:

```
btcli [options] command
```

To see the command options, enter the **btcli --help** command.

The specified command is sent to an Access Server iWRAP server (the first server at port 10101 by default) and all replies are echoed to the standard output. The application waits and prints the replies for a certain amount of time (10 seconds by default) and exits.

The iWRAP commands are described in Chapter 7.

3.2.8. serialbluetooth

It is also possible to control the first iWRAP server (at port 10101) through RS-232 with the **serialbluetooth** application.

Usage:

`serialbluetooth [options]`

To see the command options, enter the **serialbluetooth --help** command.

Basically, **serialbluetooth** takes commands from a serial port and forwards them to the iWRAP server. All the commands available through iWRAP are also available through serial port.

There are two exceptions:

1. After making an outgoing RFCOMM data call, all input from the serial port is forwarded to the data socket, not to the control socket. To close the data socket, you have to write **+++** with a 200ms pause before each character. It is not possible to have two concurrent RFCOMM calls.
2. All incoming RFCOMM calls are answered automatically. Again, to close the data socket, write **+++** as with the outgoing call.

3.3. Compact Flash Cards

Access Server functionality can be extended by using GSM/GPRS, Wi-Fi and GPS Compact Flash cards. The supported Compact Flash cards are listed in Appendix D.

3.3.1. Compact Flash GPRS Cards

The operating system automatically identifies the Compact Flash GPRS card when it is inserted. Access Server can use the GPRS card to connect to the GPRS network, or to act as an SMS gateway to send and receive SMS messages.

You can enable the GPRS mode and configure its settings, such as the SIM card's PIN code, by using the setup application or its WWW interface. For more information, see Section 2.4 and documentation for Setup ⟶ Network settings ⟶ Enable GPRS interface in Appendix B.

GPRS, when enabled, is by default only turned on when needed. If Access Server can access the Internet (or any desired address) by using the default interface `nap`, it does not activate and use the GPRS (`ppp0`) interface.

The simplest way to test the GPRS interface is to configure the default interface `nap` to use dynamic network configuration (the default) and enable GPRS through the **setup** application, then to disconnect the Ethernet cable, reboot the device with the management console enabled. After the boot, **ping** an IP address in the Internet, such as 194.100.31.45 (bluegiga.com).

The first five or so packets are lost, but after that the GPRS connection should be up. To enable the interface automatically, just enter **ping -c 20 ip-in-internet** to `/etc/rc.d/rc.local`.

> **Note:** If you also want to use the Ethernet connection, you must remove it from the default interface (`nap`) bridge and configure its network settings individually using the **setup** application while keeping the default interface network settings in their default (dynamic) state.

Using WRAP SMS Gateway Server is documented in Section 3.5.3.

If needed for some special use, the Compact Flash GPRS card can also be accessed directly from `/dev/ttyS0`, a device file which exists if the GPRS card is successfully initialized.

3.3.2. Compact Flash GPS Card

The operating system automatically identifies the Compact Flash GPS card when it is inserted. At that time, the device file `/dev/ttyS0` is created and the GPS card can be accessed by using that device with the serial port settings the GPS card uses.

The supported Compact Flash cards are listed in Appendix D.

3.3.3. Compact Flash Wi-Fi Cards

Access Server supports Prism II/III based CF Wi-Fi cards. The supported Compact Flash cards are listed in Appendix D.

By default, Access Server notices when a supported Wi-Fi card is inserted and tries to use it in the client mode, without encryption. So, if there is an open Wi-Fi Access Point in range, Access Server will automatically connect to it.

To configure Wi-Fi to the Access Point mode, or to change other Wi-Fi settings, use the setup application or its WWW interface at Setup ⟶ Network settings ⟶ Wi-Fi.

> **Note:** Older Compact Flash cards with firmware version 1.4.2 do not work in the Access Point mode. Instead, you will see an error message in the system log (`/var/log/messages`, viewable at Setup ⟶ Advanced ⟶ System Information ⟶ Show system log file).

A standard set of command line wireless utilities is provided to fine-tune your Wi-Fi configuration:

- iwconfig
- iwlist
- iwpriv

For more information on these utilities, see: http://www.hpl.hp.com/personal/Jean_Tourrilhes/Linux/Tools.ht

3.4. USB Memory Dongles and Compact Flash Memory Cards

Access Server's persistent memory storage can be extended by using an USB memory dongle or a Compact Flash memory card. These are also used by the Access Server Remote Management System (see Section 3.5.5) - each time a dongle or memory card is inserted, it is automatically mounted, and scanned for management packets, which are processed and unmounted.

To use the USB dongle or Compact Flash memory card for your own applications, the memory must be mounted manually by using command:

```
[root@wrap /]$ mount -t vfat device /mnt/usb
```

The *device* parameter is a path to the USB dongle or Compact Flash memory card filesystem device. For the first dongle inserted after a reboot, it is `/dev/sda1` if the dongle is partitioned (which often is the case) and `/dev/sda` if the dongle has no partition table. The first Compact Flash memory card is typically at `/dev/hda1`, correspondingly. If you have used several dongles after reboot, new device file names are created: `/dev/sdb1` for the second one, `/dev/sdc1` for the third one, and so on. In the case of memory cards, naming is similar, that is, the second one gets device file name `/dev/hdb1`.

Note: Always remember to unmount the memory dongle or memory card with command:

```
[root@wrap /]$ umount /mnt/usb
```

3.5. Servers

Access Server server applications are started automatically at system power-up or when an iWRAP server or the Internet services daemon needs them. The servers and their purposes are described in Table 3-2.

Server	Description
bluetooth	Access Server iWRAP Server, which is described in detail in Chapter 7.
finder	WRAP Finder Service.
obexsender	WRAP ObexSender server.
smsgw	WRAP SMS gateway server, which is described in detail in Section 3.5.3. Notice that this server is disabled by default. Use the **setup** application or the **chkconfig smsgw on** command to enable it.
watchdog	WRAP user level watchdog.
wpkgd	WRAP remote management system daemon.
crond	A daemon to execute scheduled commands. This server is configurable through the `/var/spool/cron/crontabs/root` file or the **crontab** command in the same way as any Linux crond.
ftpd	Internet File Transfer Protocol Server. You can configure this server with the **setup** application. Notice that this server is disabled by default. Use the WWW interface of the **setup** application or the **chkconfig ftpd on** command to enable it.
udhcpd	This server is a DHCP daemon for providing automatic network configuration for clients in the network. Notice that, by default, this server is only enabled for the `gn` interface, used by Bluetooth PAN Generic Networking profile.
udhcpcd	DHCP client daemon for automatic network configuration.
inetd	Internet services daemon. Notice that this server is disabled by default. Use the **setup** application or the **chkconfig inetd on** command to enable it.
httpd	Web server, which is described in detail in Section 3.5.7.
pppd	Point to Point Protocol daemon. iWRAP server uses this server. This server can be used manually over the user serial port (`/dev/ttyAT1`).
snmpd	SNMP daemon. This server is available as a separate installation packet.
sshd	SSH daemon.
syslogd	System logging daemon. This server can be configured by using the **setup** application.

Server	Description
telnetd	Telnet protocol server. Notice that this server is disabled by default. Use the **setup** application or the **chkconfig telnetd on** command to enable it.
zcip	Zero configuration networking service.
ntpd	Network Time Protocol (NTP) daemon.

Table 3-2. Access Server Servers

3.5.1. Finder

The Finder service is a small service, which listens for UDP broadcast queries from Access Server Finder applications and responses to those queries with identification information (IP address, model, serial number, etc.) about Access Server.

The **finder** command can be used to query Finder service information from Access Servers in the network. With no parameters, **finder** sends the query using the broadcast address of the default interface (`nap`). Broadcasting to networks of other interfaces can be done with `--interface` parameter, such as the zero configuration interface `nap:9`in the following example:

```
[root@wrap root]$ finder --interface nap:9
Access Server 2291 (S/N: 0402110112) (build: 3.1)
- Hostname: wrap.localdomain
- IP: 169.254.30.233 (nap:9), 192.168.161.1 (gn)
- Ethernet MAC: 00:07:80:00:03:ed
- iWRAP: 10101 00:07:80:80:0b:c3 bt1.2 (W0402110112_1)

Access Server 2291 (S/N: 0606221029) (build: 3.1)
- Hostname: wrap.localdomain
- IP: 169.254.36.138 (nap:9), 192.168.161.1 (gn)
- Ethernet MAC: 00:07:80:00:0d:44
- iWRAP: 10101 00:07:80:80:0b:c4 bt1.2 (W0606221029_1)

[root@wrap root]$
```

3.5.2. ObexSender

The ObexSender application is automatically started in Access Server. Its purpose is to receive business cards (vCards), images, or other files, and analyze their content and send files back selecting them based on configured keywords found.

ObexSender can also make an inquiry for bluetooth devices, and automatically send one or more files to all new devices found.

ObexSender can be configured with the **setup** application or by editing the `/etc/obexsender.conf` file (see Section 2.4).

For detailed instructions on using ObexSender, see Chapter 5.

3.5.3. SMS Gateway Server

WRAP SMS Gateway Server supports Nokia 20, Nokia 30, or Wavecom WMOD2 compatible GSM terminals and the supported GSM/GPRS Compact Flash cards for sending and receiving

SMS messages. By default, the Compact Flash card is used. The PIN code query of the SIM card at power-up must be disabled.

WRAP SMS Gateway Server is disabled by default. To enable it, use the **setup** application's WWW interface, as described in section Section 2.4. Enabling is done at Setup ⟶ Applications ⟶ Default bootup applications ⟶ smsgw.

WRAP SMS Gateway Server can be configured to use a modem connected to the user serial port with the setup application or its WWW interface by changing the setting at Setup ⟶ Applications ⟶ SMS gateway settings ⟶ Modem device to `/dev/ttyAT1` from the default `/dev/ttyS0`.

> **Note:** If you are using the user serial port, ensure you have Bluetooth Serial Port Profile disabled, as they share the same physical user serial port.
> **Note:** To use Nokia terminals, the device must be connected to the user serial port when the server starts up. Also, the terminal must be configured to operate in RS-232/AT command. Nokia terminals are configured with the N20 or N30 Configurator application.

For further information on using **smsgw**, see the **makesms** example in Section 6.3.1.

3.5.4. User Level Watchdog

WRAP User Level Watchdog daemon listens on UDP port 4266 for "id timeout" messages. "id" is an ASCII string, without spaces. If "timeout" equals to 0 (zero), the "id" is removed from the list of processes to wait. If "timeout" is greater than 0 (zero), the "id" is added or updated.

When there is no message for "id" received within the "timeout" seconds, the user level watchdog dies and the kernel watchdog reboots Access Server.

The **watchdog** command can be used to send messages to the watchdog daemon. This is done through command **watchdog id timeout**. For example, **watchdog test 5**.

3.5.5. Remote Management

Access Server contains simple tools that provide means for full and secure remote management of the device.

The basic remote management can be performed using the WWW Setup interface, SSH command line access, and SCP and SFTP file transfer protocols.

In addition to those, Access Server contains WRAP Remote Management System for transferring management packets over different media to Access Server and automatically sending response packets back.

The management packets (`*.wpk`) are automatically processed when they are transferred to the autoinstall directory in Access Server (`/tmp/obex` by default, but configurable with the setup application or WWW interface at Setup ⟶ Applications ⟶ wpkgd settings). The easiest way to transfer a management packet to this directory is to upload it from WWW Setup at Setup ⟶ Advanced settings ⟶ Upload a software update.

3.5.5.1. Overview

WRAP Remote Management System top level architecture is shown in Figure 3-2.

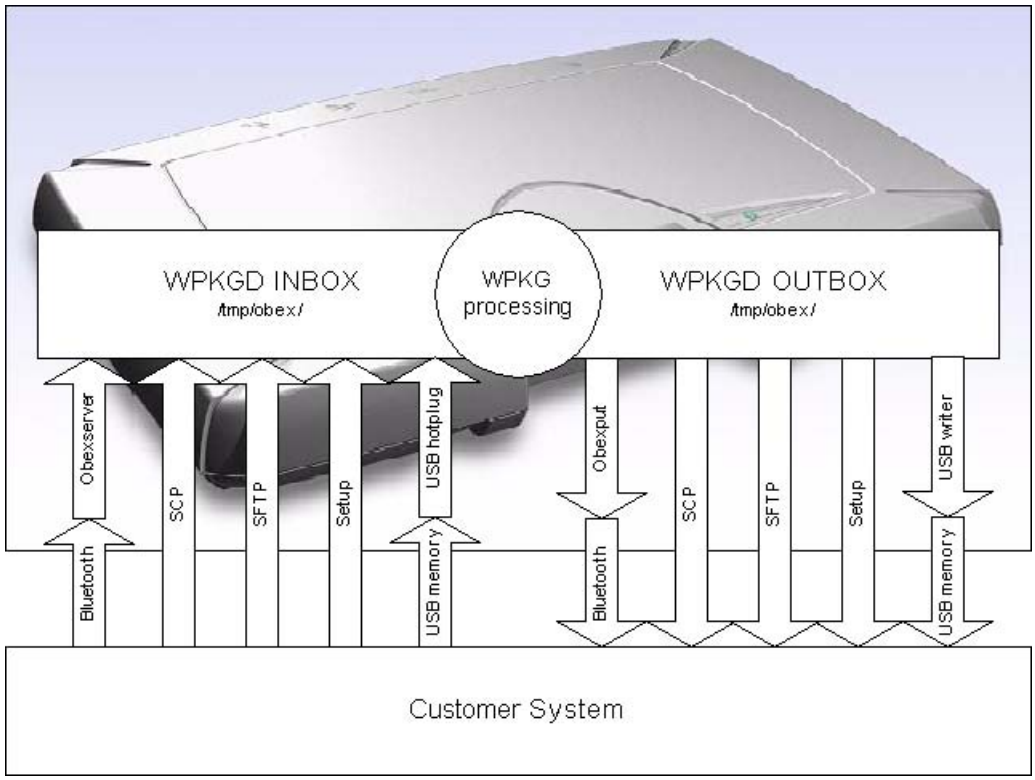

Figure 3-2. WRAP Remote Management Architecture

A management action is performed using the following procedure:

1. A customer system prepares the management packet (*.wpk).
2. The management packet is delivered to Access Server, to the packaging daemon's inbox directory. You can currently use Bluetooth, SCP, SFTP and plain FTP to do this. The packet can also be transmitted using a USB memory dongle, Compact Flash memory card or through the WWW Setup interface.
3. The Access Server packaging daemon processes the management packet, possibly generating a reply packet.
4. (Optional) The reply packet is delivered to the customer system.

3.5.5.2. Management Packet Format

- The package name must be of format `name.wpk`, where "name" can be user defined.
- Package must be a `tar` archive that is compressed with **gzip** (such as files named *.tar.gz or *.tgz).
- The package must contain a package information file called `wpkg.pif` in the package root (the file contents are described later), otherwise the built-in defaults for `wpkg.pif` are used.
- All other files, if any exist, should be data files, scripts or executables required for the management operation.

3.5.5.3. Management Packet Information File Format

The management packet information file (`wpkg.pif`) consists of tags and their data, described here:

%wpkg-version: 2

Contains information for version checking. 2 is currently the only supported version. It is also the default value.

%wpkg-prepare: [command line[s]]

One or more commands (all commands are lines until the next tag is interpreted as a command line) to execute. Commands may contain parameters, redirections and job control as well.

The built-in default value for this is **/usr/bin/dpkg -i *.deb || echo ERROR: Installation failed.**. This enables the special case of creating .wpk packets from .deb packets simply with **tar czf foo.wpk foo.deb**. (`wpkg.pif` is not needed in this special case).

%wpkg-reply: method

This value indicates where the generated reply packet is sent. By default, it is sent to where it came from. Possible values are:

- default
- file:///path/filename
- scp://remote:file
- objp://bdaddr/
- none

%wpkg-format: type

This value indicates what kind of a reply packet will be generated. Possible values are:

- ascii (this is the default value, everything echoed by the prepare-section will be sent).
- tgz (all files in the current directory will be sent).
- vcf (same as ascii, but assume it is a vCard).
- vmg (same as ascii, but assume it is a vMessage).
- vnt (same as ascii, but assume it is a vNote).
- vcs (same as ascii, but assume it is a vCalendar).
- html (same as ascii, but assume it is HTML).

%wpkg-auth: auth

Optional authentication string required by wpkgd.

3.5.5.4. Management Operation Example: Hello World

See below for the simplest example of `wpkg.pif`:

```
%wpkg-version: 2
%wpkg-prepare:
```

```
echo Hello world
```

This will generate a reply packet containing text "Hello world". You can generate the wpk file simply by giving the command **tar czf hello.wpk wpkg.pif**.

3.5.5.5. Management Operation Example: Software Update

See below for a more complex example of **wpkg.pif**:

```
%wpkg-version: 2
%wpkg-prepare:
FOO=`pwd`
cd /
tar xzf ${FOO}/files.tar.gz
echo Done.
```

This example will extract files from the included `files.tar.gz` file. You can generate the wpk file with command **tar czf update.wpk wpkg.pif files.tar.gz**.

3.5.5.6. Management Operation Example: IPQUERY

In this example, we build a simple packet that can be used with a Bluetooth enabled phone to retrieve the IP address of an Access Server. File `wpkg.pif` reads:

```
%wpkg-version: 2
%wpkg-format:  vcf
%wpkg-prepare:

ipaddr() {
echo `ifconfig nap | grep "inet addr" | awk -F [:] \
  \\{print\\$2\\} | awk \\{print\\$1\\}`
}

serialno() {
echo `wrapid | grep Hardware | awk \\{print\\$5\\}`
}

echo -e "BEGIN:VCARD\r"
echo -e "VERSION:2.1\r"
echo -e "N:`serialno`\r"
echo -e "TEL:`ipaddr`\r"
echo -e "URL:`hostname`\r"
echo -e "END:VCARD\r"
```

This example will send the reply back as a vCard (contact card). Please note that you have to include all required vCard formatting by yourself. You can generate the wpk file simply giving the command **tar czf ipquery.wpk wpkg.pif**.

To use this example, send the file `ipquery.wpk` to the inbox of your Bluetooth phone. Check that you have Bluetooth enabled in the phone. Then, from the phone's inbox, send the file `ipquery.wpk` over Bluetooth to Access Server.

3.5.5.7. Management with USB Memory Dongle or Compact Flash Memory Card

When an USB memory dongle or Compact Flash memory card is inserted, Access Server automatically tries to mount it (using VFAT type). If the mount is successful, Access Server scans the root for `*.wpk` packets. If a packet is found, the WRAP Package daemon handles it. Optional reply packets are saved back to the root folder (unless otherwise stated in the **%wpkg-reply** tag).

3.5.6. FTP

If you enable the FTP server, users can use it to log in anonymously to the `/tmp/obex` directory with download access or as **`root`** with password **`buffy`** to the root directory with full access. The password and other settings can be changed on Access Server with the **setup** application or by editing the `/etc/ftpd.conf` file (see Section 2.4).

> **Note:** Do not enable FTP because it is insecure. Use SSH (SCP or SFTP) instead. A commonly used client with a graphical user interface is, for example, WinSCP (http://winscp.net/).

3.5.7. Web Server

The integrated web server in Access Server supports HTTP/1.0 methods GET and POST, and has light user authentication capabilities. The content can be either static or dynamic - the WWW server is CGI/1.1 compatible.

The web server is always running and the content (http://wrap-ip-address/) is located in the `/var/www/html/` directory in Access Server's file system.

The web server is configured to protect the WWW Setup interface with a username and password. The default username and password can be changed as instructed in Section 2.4. For further information about using the web server for your own applications, see the web examples in Section 6.3.1.

3.5.8. SNMP

A separate software update package is available from Bluegiga Techforum (http://www.bluegiga.com/techforum/). This update adds the Net-SNMP suite of applications to Access Server. The current Net-SNMP implementation for Access Server is limited and will be extended in the future. However, it can be used to poll the basic status of Access Server.

Configuration details can be found and altered in configuration file `/etc/snmp/snmpd.conf`, which is accessible as described in Section 2.4.

For more information about the Net-SNMP suite, see http://net-snmp.sourceforge.net/

3.5.9. OpenVPN

A separate software update package is available from Bluegiga Techforum (http://www.bluegiga.com/techforum/). This update adds the OpenVPN™, a full-featured SSL VPN solution, to Access Server.

For detailed instructions on using OpenVPN with Access Server, see Section 9.4.

For more information about the OpenVPN™, see http://openvpn.net/.

3.5.10. SSH

By default, users can use SSH to log in (or SCP and SFTP to transfer files) as user **root** with password **buffy**. The password can be changed on Access Server by using command **passwd** or with the **setup** application.

3.5.11. Telnet

If you enable telnet, users can log in over telnet as user **root** with password **buffy**. The password can be changed on Access Server using the command **passwd** or with the **setup** application.

Note: Do not enable telnet because it is insecure. Use SSH instead.

3.5.12. NTP

The **ntpd** service uses the standard Network Time Protocol (NTP) to keep Access Server system time automatically in sync using a random selection of eight public stratum 2 (NTP secondary) time servers. The service is also configured to answer NTP requests from other devices.

The NTP server configuration can be altered by editing its configuration file `/etc/ntpd.conf`.

3.6. Utilities

Access Server is basically a small Linux system. Whether logged in from the management console or with SSH, your shell session starts as the root user in the root directory. After that, you have the option to use most of the standard Linux utilities, briefly listed and described in Table 3-3. Most of the commands have a small built-in usage help that can be seen by executing the command with the **-h** or **--help** parameter.

Application	Description
adduser	This command add user to the system.
arping	This command pings hosts by ARP requests/replies.
awk	Pattern scanning and processing language.
b2b_class1	WRAP baseband module control script (set basebands to class 1).
b2b_class2	WRAP baseband module control script (set basebands to class 2).
b2b_class3	WRAP baseband module control script (set basebands to shortest possible range).
basename	Strip directory and suffix from file names.
bash	Bourne-Again SHell.
btcli	WRAP iWRAP Server Command Line Interface utility.
btproxy	WRAP iWRAP Proxy for Access Servers (test revision).
bunzip2	Decompress bzip2-compressed files.
bzcat	Decompress bzip2-compressed files to stdout.
cardctl	Monitor and control the state of PCMCIA sockets.
cat	Concatenate files and print on the standard output.

Application	Description
chat	Automated conversational script with a modem.
chgrp	Change group ownership.
chkconfig	Updates and queries runlevel information for system services.
chmod	Change file access permissions.
chown	Change file owner and group.
chroot	Run command or interactive shell with special root directory.
clear	Clear the terminal screen.
cmp	Compare two files.
cp	Copy files and directories.
cpio	Copy files to and from archives.
crontab	Maintain crontab files for individual users.
cut	Remove sections from each line of files.
date	Print or set the system date and time. Notice that the date command does not store the date into the battery powered real time clock. Use the hwclock application instead.
dd	Convert and copy a file.
deluser	Delete a user from the system.
df	Report file system disk space usage.
dfu	WRAP baseband module firmware upgrade tool.
dialup	WRAP iWRAP helper application.
dirname	Strip non-directory suffix from file name.
dmesg	Prints or controls the kernel ring buffer.
dpkg	A medium-level package manager for (.deb) packages.
dpkg-deb	Debian package archive (.deb) manipulation tool.
du	Estimate file space usage.
dump_cis	Retrieves and parses the Card Information Structures for inserted PCMCIA devices, or optionally, parses CIS information from a file.
dun	WRAP iWRAP helper application.
egrep	Print lines matching a pattern.
encode_keychange	Produce the KeyChange string for SNMPv3.
env	Run a command in a modified environment.
expr	Evaluate expressions.
false	Do nothing, unsuccessfully.
fgrep	Print lines matching pattern.
find	Search for files in a directory hierarchy.
free	Display the amount of free and used memory in the system.
ftp	Internet file transfer program.
gdbserver	Remote server for GDB debugger. Available in a separate software package.

Application	Description
getty	Opens a tty, prompts for a login name, then invokes `/bin/login`.
grep	Print lines matching a pattern.
gunzip	Expand gzip compressed files.
gzip	Compress files into gzip format.
head	Output the first part of files.
hexdump	A filter which displays the specified files, or the standard input, if no files are specified, in a user specified format.
hostid	Print out a unique 32-bit identifier for the machine (not yet implemented).
hostname	Show or set the system's host name.
hwclock	Query and set the hardware clock.
id	Print information for username or current user.
ide_info	IDE device information.
ifconfig	Configure a network interface.
ifport	Select the transceiver type for a network interface.
ifuser	Checks to see if any of the listed hosts or network addresses are routed through the specified interface.
insmod	Loads the specified kernel modules into the kernel.
ip	TCP/IP interface configuration and routing utility.
iptables, ip6tables	IP packet filter administration.
kill	Terminate a program.
killall	Kill processes by name.
ln	Make links between files.
logger	Make entries into the system log.
login	Sign on.
ls	List directory contents.
lsmod	List loaded modules.
md5sum	Compute and check MD5 message digest.
mkdir	Make directories.
mknod	Make block or character special files.
mktemp	Make a temporary file name (unique).
modprobe	High level handling of loadable modules.
more	File perusal filter for crt viewing.
mount	Mount a file system.
mv	Move (rename) files.
net-snmp-config	Net-SNMP tool.
nslookup	Queries the nameserver for IP address of given host.
ntpd	Network Time Protocol NTP daemon.

Application	Description
obexbrowser	The WRAP obexbrowser. A command line OBEX client interface.
obexget	The WRAP OBEX tool for retrieving a file from a remote device with ObjP/FTP support.
obexput	The WRAP OBEX tool for sending a file to a remote device with ObjP/FTP support.
pack_cis	Convert a text description of a PCMCIA Card Information Structure (CIS) to its packed binary representation.
passwd	Update a user's authentication token(s).
picocom	Minimal dumb-terminal emulation program. Available in a separate software package.
pidof	Find a process ID of a running program.
ping, ping6	Send ICMP ECHO_REQUEST packets to network hosts.
ps	Report process status.
pwd	Print the name of the current/working directory.
rb, rx, rz, sb, sx, sz	Xmodem, Ymodem, Zmodem file receive and send.
rdate	Get and possibly set the system date and time from a remote HOST.
reboot	Reboot the system.
renice	Alter the priority of running processes.
reset	Resets the screen.
rm	Remove files or directories.
rmdir	Remove empty directories.
rmmod	Unload loadable modules.
route	Show / manipulate the IP routing table.
scp	Secure copy (remote file copy program).
scsi_info	SCSI device description tool.
sed	A Stream EDitor.
setup	The WRAP Setup Application. See Section 2.4.
sftp	Secure file transfer program.
sleep	Delay for a specified amount of time.
snmp*	Set of standard SNMP command line applications.
sort	Sort lines of text files.
ssh, slogin	OpenSSH SSH client (remote login program).
ssh-keygen	SSH authentication key generation, management and conversion.
strace	Utility to trace system calls and signals. Available in a separate software package.
strings	Display printable strings in binary file.
stty	Change and print terminal line settings.
su	Run a shell with substitute user and group IDs.
sulogin	Single-user login.

Application	Description
supportinfo	Output collectively all the system status and configuration information.
sync	Flush filesystem buffers.
tail	Output the last part of files.
tar	Tar archiving utility.
tcpdump	Utility for dumping traffic on a network. Available in a separate software package.
telnet	User interface to the TELNET protocol.
test	Check file types and compare values.
time	Run command and display its resource usage information when finished.
top	Provides a view to processor activity in real time.
touch	Change file timestamps.
tr	Translate or delete characters.
traceroute	Trace the route that IP packets take on their way to the host.
true	Do nothing, successfully.
tty	Print the file name of the terminal connected to standard input.
uartmode	WRAP Uartmode: Change the mode of the user serial port (DTE or DCE).
umount	Unmount file systems.
uname	Print system information.
uniq	Remove duplicate lines from sorted lines.
unzip	List, test, and extract compressed files in a ZIP archive.
uptime	Tell how long the system has been running.
usleep	Sleep some number of microseconds.
uudecode	Decode a file create by uuencode.
uuencode	Encode a binary file.
wc	Print the number of bytes, words, and lines in files.
vi	A text editor.
wget	A utility to retrieve files from the World Wide Web.
wrapfinder	Finds other Access Servers in the network.
wrapid	Access Server identification program. Shows build and hardware configuration information.
which	Shows the full path of (shell) commands.
whoami	Prints the user name associated with the current effective user id.
zcat	Expand gzip compressed files to the standard output.
zcip	Zero Configuration Networking application.
xargs	Build and execute command lines from the standard input.

Table 3-3. Access Server Utilities

3.7. Real Time Clock

The system clock is read from the battery operated real time clock during boot. The time between the system time and the real time clock is automatically synchronized when the system is rebooted using the **reboot** command. Synchronizing can also be done using the **hwclock --systohc --utc** command. Give command **hwclock --help** for more information about the **hwclock** utility.

3.8. Time Zone

The default time zone in Access Server is UTC. You can change the timezone by replacing the file `/etc/localtime` with the correct file from your desktop Linux system (using your `/etc/localtime` or a desired zone from `/usr/share/zoneinfo`).

3.9. System Re-Install and Upgrade

Access Server can be re-installed with the latest software version. The latest software updates and instructions are available at http://www.bluegiga.com/techforum/.

Most of the software updates are delivered as a `wpk` file.

The easiest way to install the latest software version is:

1. Start Access Server.
2. Copy the `wpk` file or files to an empty USB memory dongle.
3. Insert the dongle in Access Server
4. One or several LEDs will turn on, and after 10-60 seconds they will all turn off.
5. Remove the dongle and reboot Access Server.
6. You have now successfully upgraded Access Server.

See Section 3.5.5 for detailed descriptions of other options and how to create your own `wpk` files.

Chapter 4. SPP-over-IP

SPP-over-IP is a special functionality of iWRAP Bluetooth servers running in Access Servers. It offers a transparent way to transmit data from Bluetooth Serial Port Profile (SPP) enabled devices to server computers or PCs. Several transport medium are supported, such as Ethernet, Wi-Fi or and GPRS.

4.1. How SPP-over-IP Works

The SPP-over-IP application enables transparent data transfer between any Bluetooth Serial Port Profile (SPP) complaint device and a server, laptop or desktop connected to the same network. This enables plug n' play connectivity from a Bluetooth network to any standard TCP/IP based network. See Figure 4-1 for an overview of the application and a brief introduction to its functionality.

Features of SPP-over-IP are:

- Access Server 2291 supports 7 incoming SPP connections.
- Access Server 2293 supports 21 incoming SPP connections.
- SPP-over-IP can be used over Ethernet, Wi-Fi or GRPS networks.
- SPP-over-IP also works over Bluetooth Personal Area Networking (PAN) connections, so not all Access Servers need to be physically (cable) connected to the TCP/IP network, but some Access Servers can linked using the Bluetooth PAN connection. This is referred to as *repeater* operation.
- If SPP-over-IP application cannot open the TCP connection to defined IP address and port, the SPP connection will not be accepted.
- If the TCP server on PC is closed, all SPP connections will be closed as well.
- When Access Server is in its default configuration, it tries to enable sniff power saving mode on all idle Bluetooth connections to minimize power consumption.
- SPP-over-IP can also be used to opposite direction, i.e. Access Server opens the Bluetooth connections to dedicated Bluetooth devices. See Section 4.1.4 for more details.
- SPP-over-IP can also be combined with the Tactical Software's Serial/IP® software. Serial/IP software converts automatically TCP connections to virtual COM ports on the host PC, so legacy applications utilizing COM-ports instead of TCP/IP can also be used.

4.1.1. Standard Operation

With the standard configuration, SPP-over-IP works as described below:

- Listens for incoming Serial Port Profile (SPP) connections
- Takes control of all incoming connections
- Opens a TCP connection to the defined IP address and TCP port
- Forwards all incoming data from the SPP device to the established TCP connection and vice versa

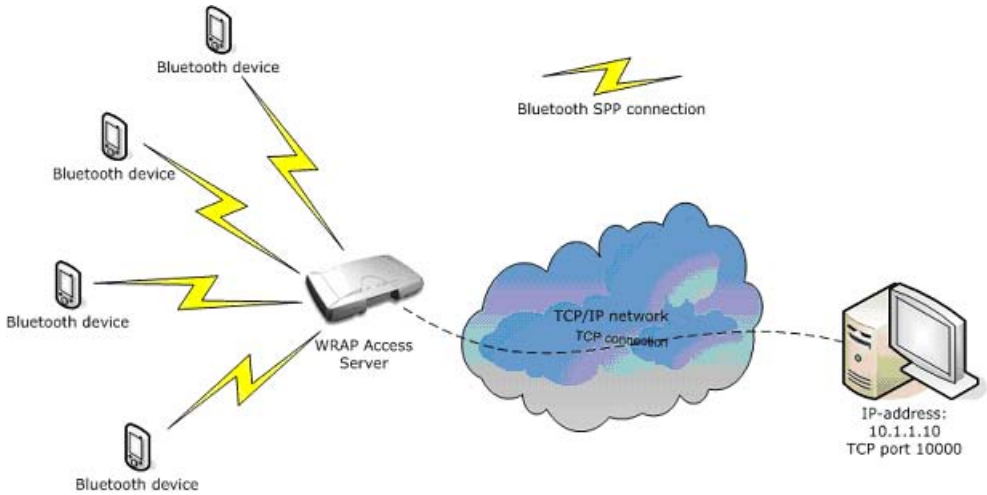

Figure 4-1. SPP-over-IP Network Architecture

All the server computer needs to do is to listen for incoming TCP connections from Access Server to a specified TCP port and receive/send the application data.

4.1.2. Repeater Operation

The SPP-over-IP application can also be used in a so-called repeater mode. This feature is useful when all Access Servers can not be directly connected to the TCP/IP network, but they can be connected to other Access Servers by using Bluetooth PAN-connection. PAN enables transmitting TCP/IP packets wirelessly over Bluetooth. The figure below illustrates this configuration:

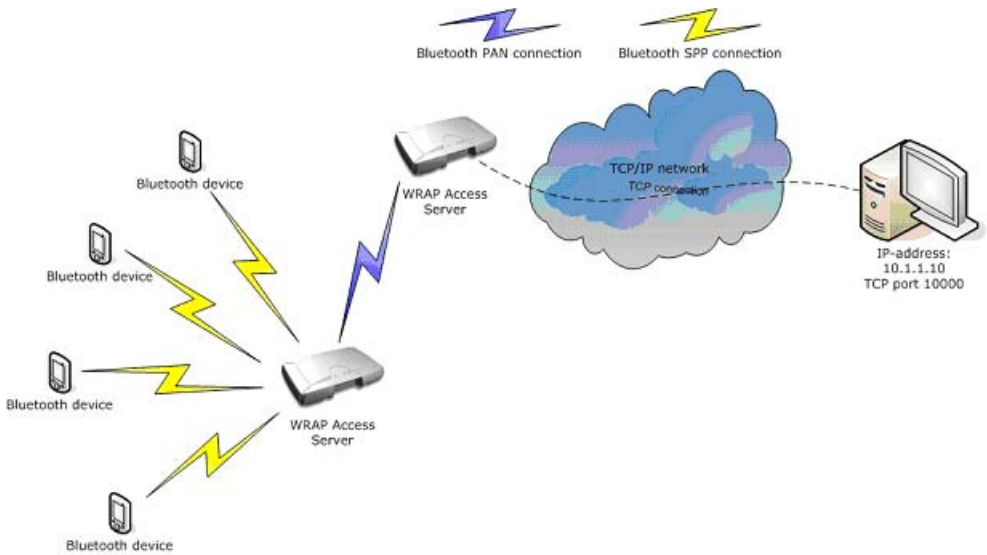

Figure 4-2. Repeater Mode in SPP-over-IP

4.1.3. SPP-over-IP over GPRS

SPP-over-IP software can also be used over GPRS instead of wired Ethernet connection. This

requires that Access Server is equipped with a working GSM/GPRS compact flash card. See Appendix D for supported cards.

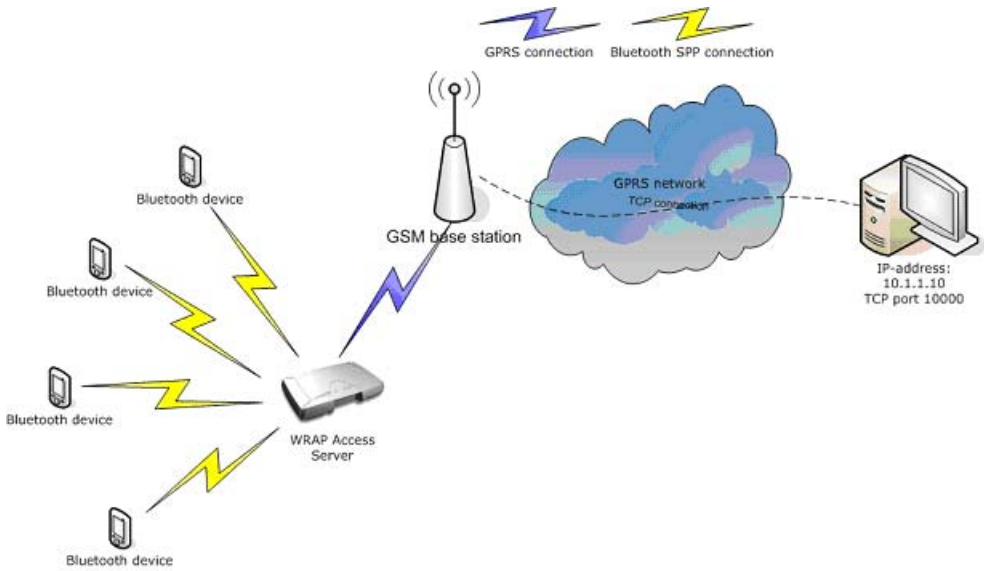

Figure 4-3. SPP-over-IP over GPRS

Notice when using GPRS:

- Data upload rate is around 8-12kbps (depending on GPRS card)
- Data download rate is around 32-48kbps (depending on GPRS card)
- Data transmission delays can be very high, sometimes even seconds
- GPRS connection may be unreliable and break easily. This should be taken account when designing the system. If GPRS connection breaks, all the TCP and Bluetooth connections will also be closed.

4.1.4. Opening Connections from Access Server

In the basic SPP-over-IP use case, Access Server is in passive mode and only accepts incoming connections. It is however possible to implement a system where Access Server opens the Bluetooth connections to the defined static Bluetooth devices or, alternatively, on ad-hoc basis.

In this case, special software must be developed for Access Server, which handles the outgoing connections and decides where they are opened to. This software can be developed with the Access Server Software Development Kit (SDK). The software can be written with C, C++ or standard Linux scripts.

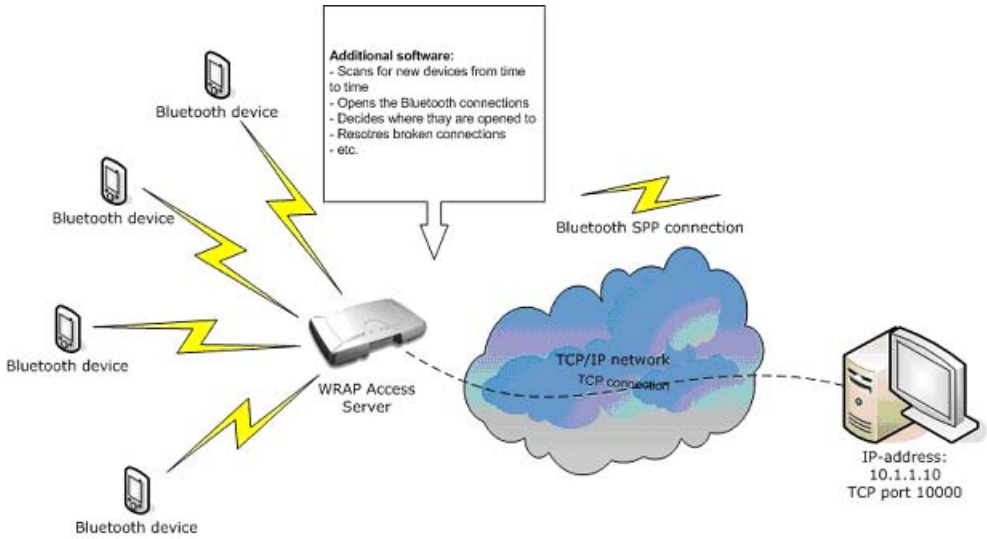

Figure 4-4. Access Server Opening the Connections

4.1.5. SPP-over-IP and COM Ports

SPP-over-IP can also be used together with Tactical Software's Serial/IP® software. Serial/IP software simply converts the TCP connections into virtual COM ports on the host computer. This is very useful in applications, which do not have support for TCP/IP but support COM ports instead.

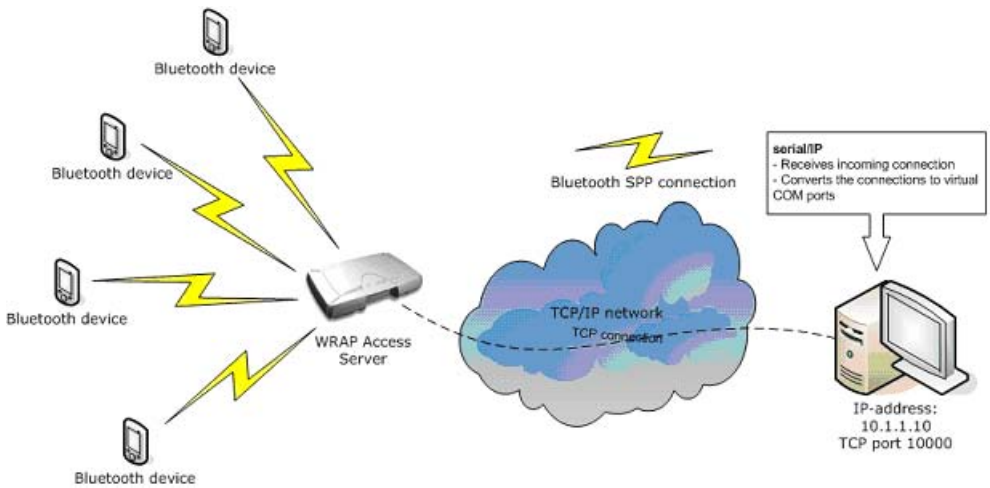

Figure 4-5. SPP-over-IP with Serial/IP

An evaluation version of Serial/IP can be downloaded from: http://www.tacticalsoftware.com/products/serialip.htm

4.2. Configuring SPP-over-IP

This chapter briefly instructs you to configure SPP-over-IP to work in different network setups or use cases.

4.2.1. Preparations

SPP-over-IP is easiest to configure through WWW setup, which allows you to access all the necessary configurations.

First, you must figure out Access Server's IP address (if it is connected to a TCP/IP network). This is easiest to do with the WRAPFinder software:

1. Start the WRAPFinder software
2. Scan your network for available Access Servers
3. Choose the correct Access Server
4. Press the Connect button

   Your web browser opens the WWW setup of the selected Access Server.

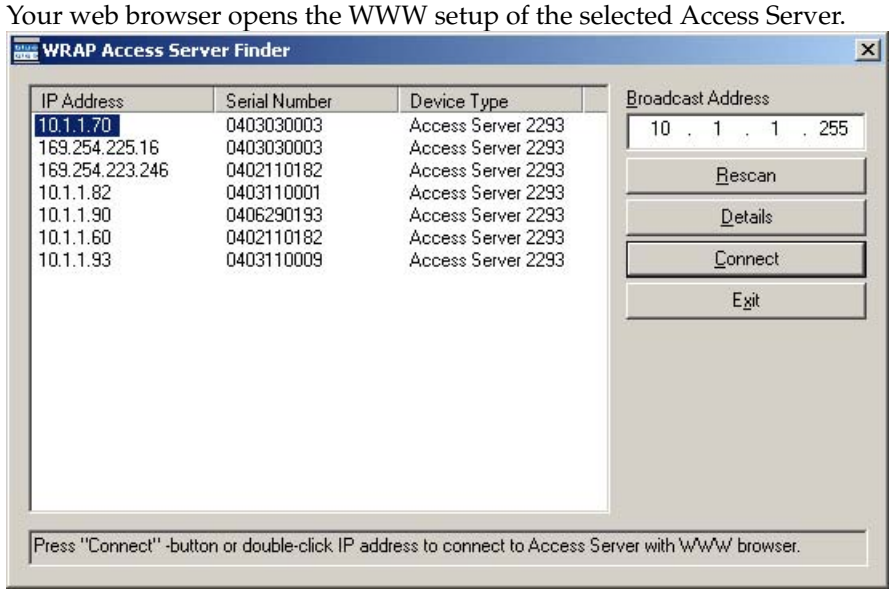

Figure 4-6. WRAPFinder

5. Once the browser window has opened, click the Setup link

Figure 4-7. WWW Setup Login

6. Type in you *user name* and *password* and you get access to the main view of the setup:

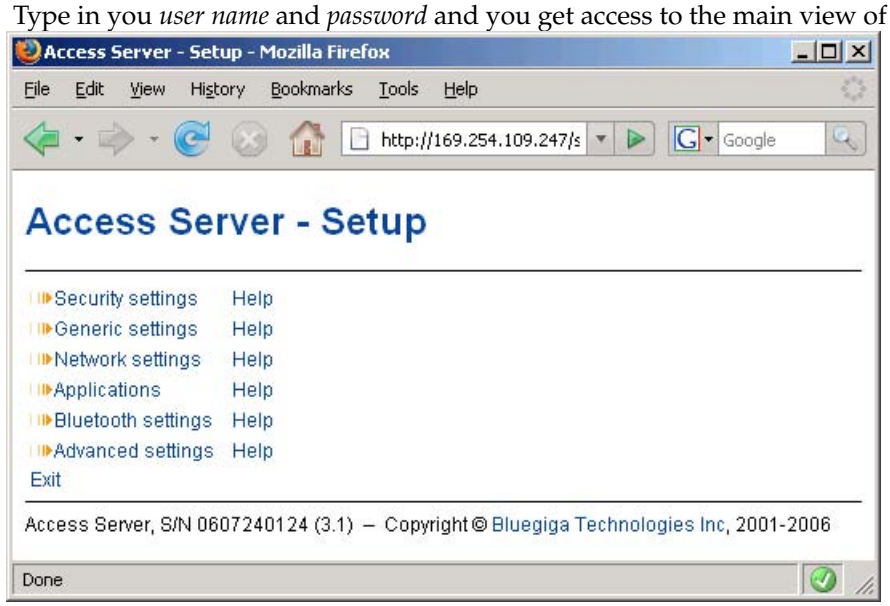

Figure 4-8. WWW Setup Main View

Note: The "basic" Bluetooth Serial Port Profile must be disabled for SPP-over-IP to work. By default, this is the case. You can verify it by checking that **serialport** service (which implements the profile) is off in WWW Setup ⟶ Applications ⟶ Default startup Applications (see Figure 4-9).

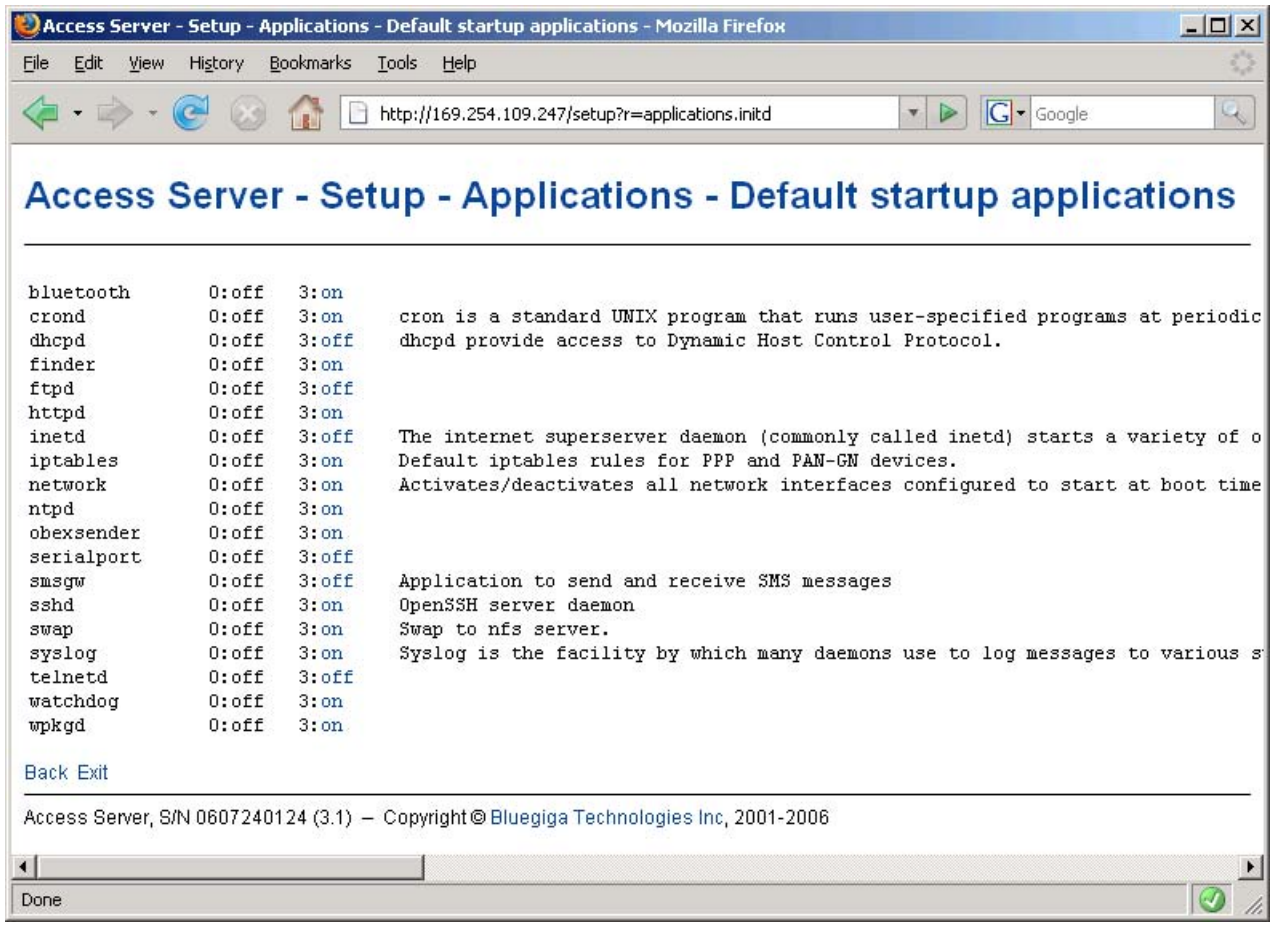

Figure 4-9. Checking that Bluetooth Serial Port Profile is disabled.

4.2.2. Preparations

SPP-over-IP settings are modified in iWRAP servers' configuration file `/etc/bluetooth.conf`which can be edited by navigating in WWW Setup to Setup ⟶ Bluetooth settings ⟶ Edit startup script.

To enable SPP-over-IP, add lines similar to following to the end of that file (lines starting with **#** are comments which can be left out):

```
# Forward incoming connection to IP 192.168.42.99 socket 7444
SET BLUETOOTH LISTEN 1 192.168.42.99:7444

# Add SDP record for Serial Port Profile
SDP ADD SPP 1 "SPP-over-IP"
```

In the example configuration above, RFCOMM channel 1 is used by the SPP-over-IP service. You can, however, use any other free channel as well. The RFCOMM channel must be same in both **SDP ADD** (see SDP ADD for details of command syntax) and **SET BLUETOOTH LISTEN** (see Table 7-1 for details of command syntax) configuration commands.

The text "SPP-over-IP" is the name of the service shown in Bluetooth service discovery. Normally, there should be no need to specify a different name, but nobody forces you to use "SPP-over-IP".

In the example, connections are forwarded to a server listening for incoming connections to TCP port 7444 in host with IP address 192.168.42.99. You must change these to match your system.

See Figure 4-10 for WWW Setup example of configuration.

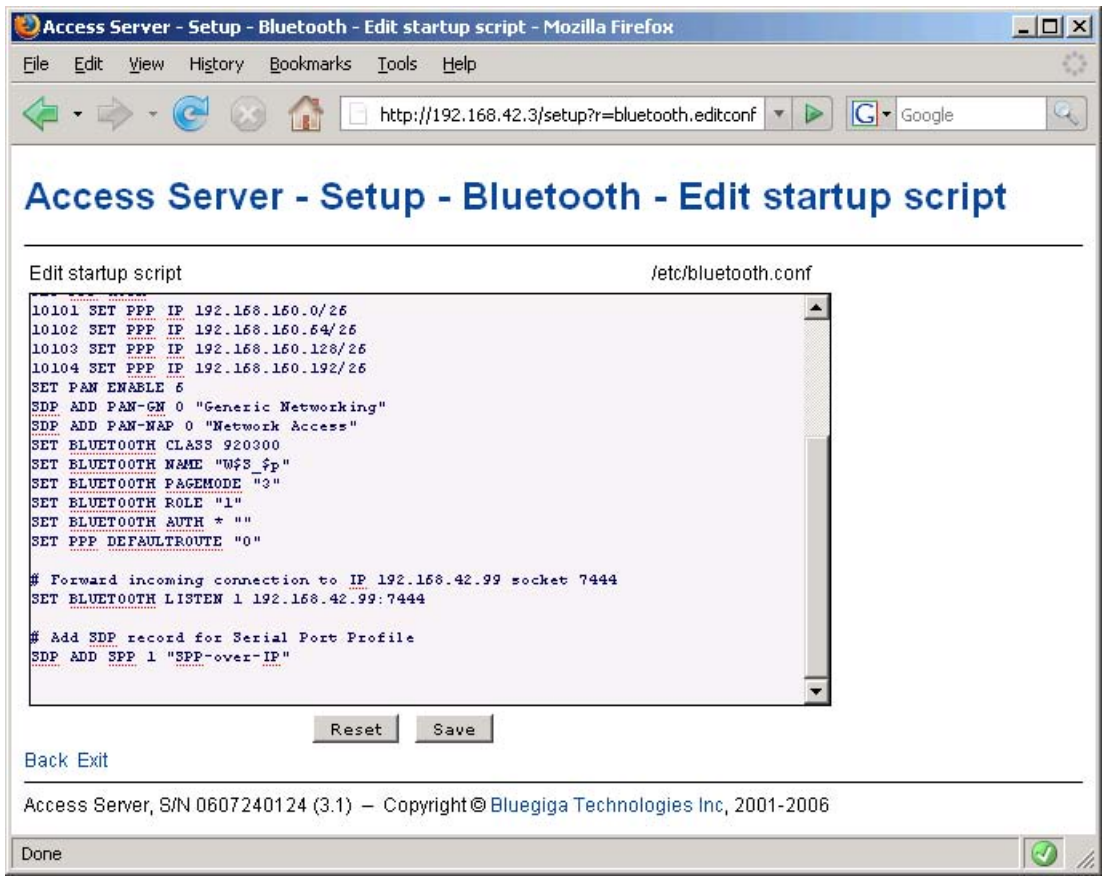

Figure 4-10. SPP-over-IP Configuration Made over WWW Setup

Once you have done your configuration, press the Save button and restart the server so that the settings take place.

4.2.3. Repeater Configuration

If you want to configure Access Server also to act as a repeater (see Figure 4-2) you must make some additional configurations. Add the line below to your Bluetooth startup script (line starting with **#** is comment which can be left out):

```
# Automatically connect to Access Server with PAN-NAP enabled
SET CONTROL AUTOEXEC CALL 00:07:80:bf:01 PAN-NAP
```

You must replace the Bluetooth address used in the example (00:07:80:80:bf:01) with the Bluetooth address of the Access Server, on which you want to receive the PAN connection.

> **Note:** The server receiving the PAN connection must have the PAN-NAP profile enabled. This is by default not the case, so in setup or its WWW interface, ensure that the setting at ⟶ Bluetooth settings ⟶ Bluetooth profiles ⟶ Enable PAN network access point profile says yes. No other configuration is needed. See Section 3.2.5 for more information on PAN profiles.

The Bluetooth PIN codes must be the same in both Access Servers.

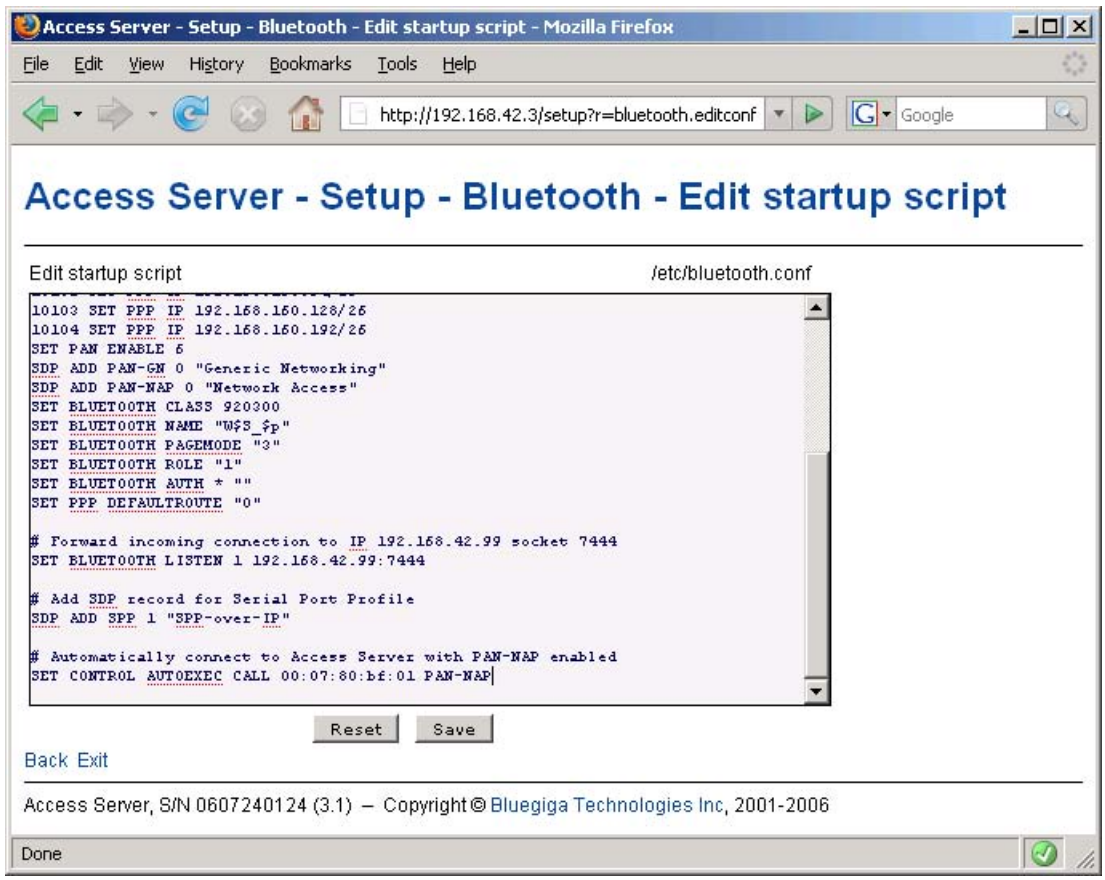

Figure 4-11. Repeater Configuration

4.2.4. Wi-Fi Configuration

If Access Servers must be connected to Wi-FI (WLAN) instead of physical Ethernet connection, you also need to make additional configurations through the WWW setup.

See Section 3.3.3 for more information.

4.2.5. GPRS Configuration

If Access Servers must be connected to GPRS network instead of physical Ethernet or Wi-Fi connection, you also need to make additional configurations through the WWW setup.

See Section 3.3.1 for more information.

Chapter 5. Obexsender

Obexsender is one of the built-in applications in Access Server. It is dedicated to Bluetooth proximity marketing, content distribution, location based services, and much more. Access Server plus Obexsender provide the user with a ready platform to start content distribution including all the necessary Bluetooth functions from discovering the devices to transmitting the content. The user needs to only focus on what, when, and to whom to send the content - rest is taken care of by Access Server and Obexsender.

The figure below illustrates a simplified Obexsender network:

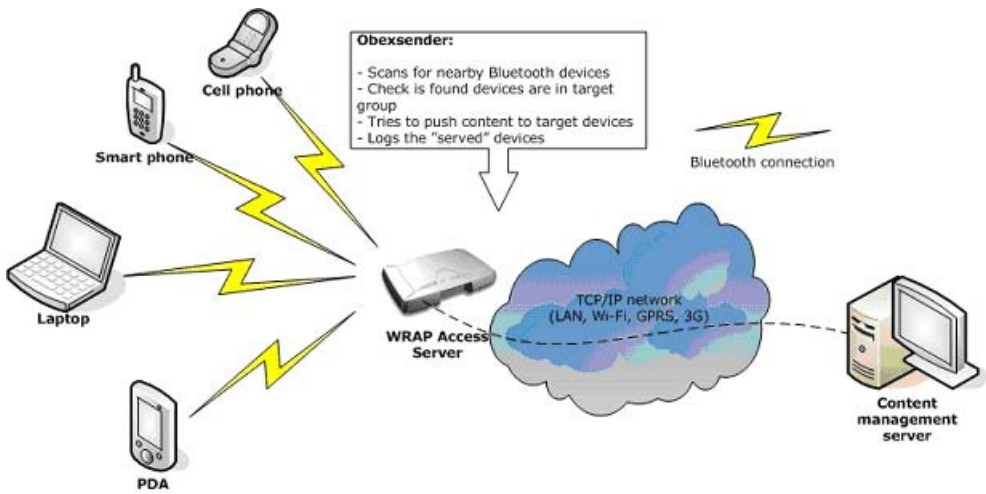

Figure 5-1. Simplified Obexsender network

5.1. Key Features

- Automatic device discovery and content push over a Bluetooth connection
- 18 simultaneous Bluetooth connections with one Access Server
- Upload speed even up to 75KB/sec with Bluetooth 2.0+EDR
- Content can be stored locally - with external memory even up to 2GB space
- Wide networking support: Bluetooth, Ethernet, Wi-Fi, GPRS and EDGE
- Secure remote connections over a Virtual Private Networking
- Remote file system support
- Lots of filtering options, such as device type, or distance from access server
- Extensive logging
- Interaction between several Access Servers
- Content time stamping

5.2. Use Cases

This chapter describes some possible Obexsender use cases.

5.2.1. Content Push

This is the standard functionality in Obexsender. In content push mode, Obexsender is scanning for devices and pushing it to clients who belong to the target group (not opted out by filtering).

Figure 5-2. Obexsender Use Case: Content Push

5.2.2. Content Pull

Obexsender can also be configured into a content pull mode. In this mode, the transaction is initiated by the user. The user can send any file to the server or alternatively a file containing some specific string such as "MP3" or "NOKIA N73". The server parses the received file and as a response pushes a corresponding file to the user if such exists.

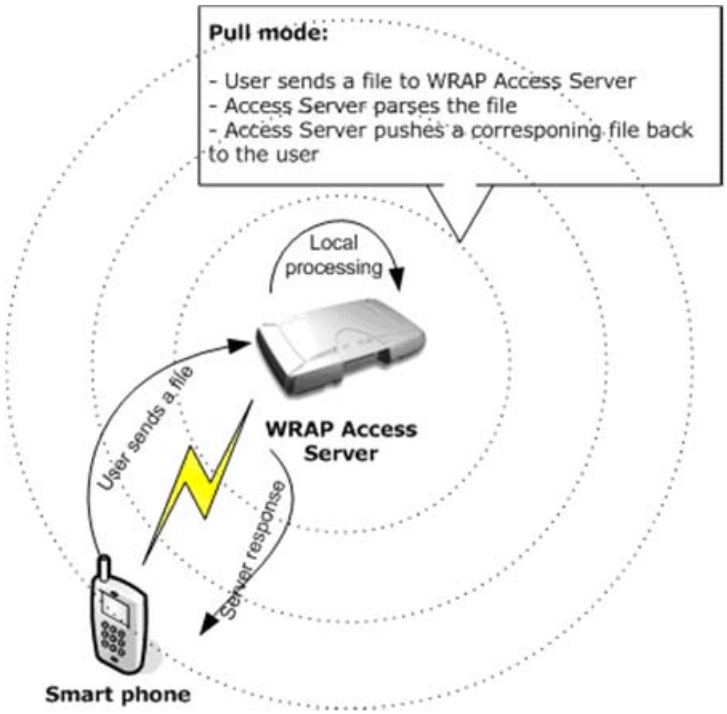

Figure 5-3. Obexsender Use Case: Content Pull

5.3. Configuration

This chapter contains instructions from the most basic Obexsender configuration to the more advanced use cases.

5.3.1. Getting Started

The easiest and fastest way to configure Obexsender is through the WWW setup. To do this, your Access Server must be connected to the same network as your PC or, alternatively, you can also use a direct Ethernet cross cable or a Bluetooth PAN connection (see Section 3.2.5 for instructions on how to enable PAN). By default, Access Server uses DHCP, so if you connect it to your LAN, it must support DHCP as well.

1. Once you have successfully connected Access Server, start the "WRAPFinder" software. WRAPFinder lists all the Access servers in the same network as your PC.

   If Access Server does not show immediately, you may need to push the Rescan button a couple of times.

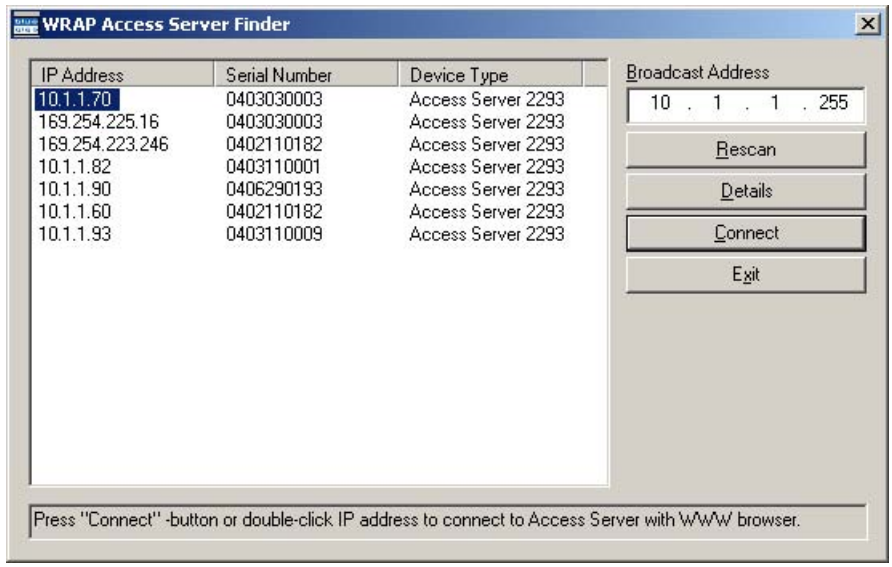

Figure 5-4. WRAPFinder

2. Next, select the correct Access Server and press the Connect button in the WRAPFinder user interface. An internet browser window opens with the Access Server IP address in the address bar.

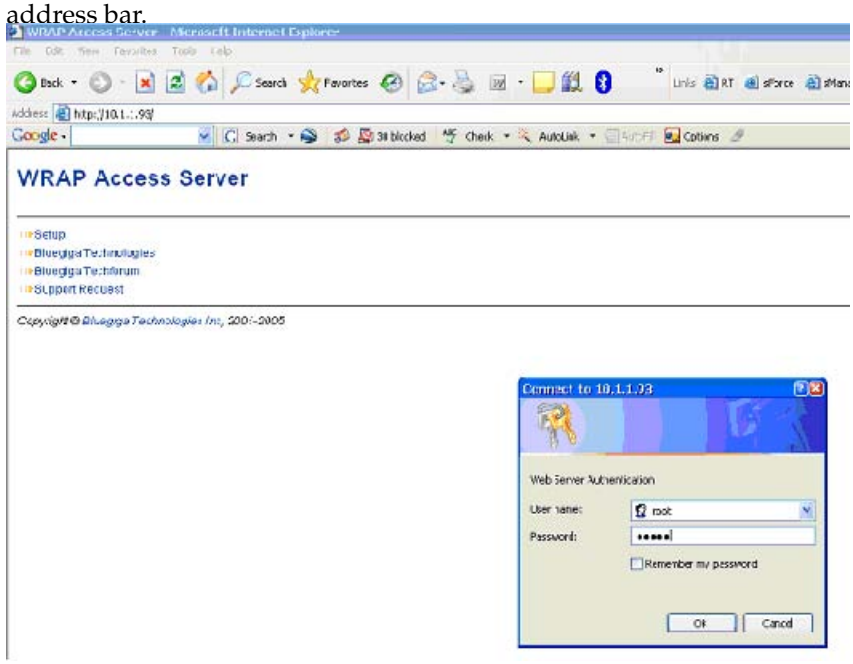

Figure 5-5. Access Server WWW Setup

3. Click the Setup link. A login screen is opened. Enter a correct user name and password.

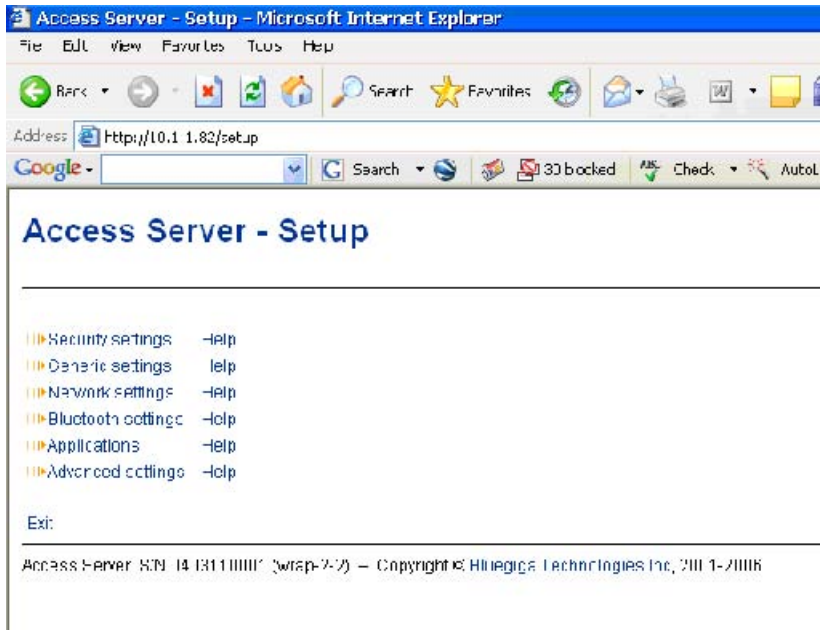

Figure 5-6. WWW Setup Main Page

4. After a successful login, you get access to the WWW setup main page.
   Note: At this point, you should check your access server software version. Obexsender works only with software version 2.2.0 and newer. At the bottom of the screen you should see a line saying:
   `Access Server, S/N 0511170051 (wrap-2-2) - Copyright © Bluegiga Technologies Inc, 2001-2006`

   If the version is older than "wrap-2-2", you must first update your Access Server. Latest software releases and instructions can be found from www.bluegiga.com/techforum/

5.3.2. Updating Obexsender

If you have software version 2.2.0 in your Access server, you need to update Obexsender to the latest version. It offers many new, useful and necessary features that include:

- Retry delay, scan delay and reply delay
- Dump delay
- Possibility to save incoming files, i.e. remote requests
- Watchdog support
- Regexp and Unicode support
- Other minor bug fixes and improvements

The rest of the manual concentrates on the latest Obexsender, but it also covers all the features offered by previous Obexsender. The main menu of latest (at the time of writing) Obexsender is shown in Figure 5-7

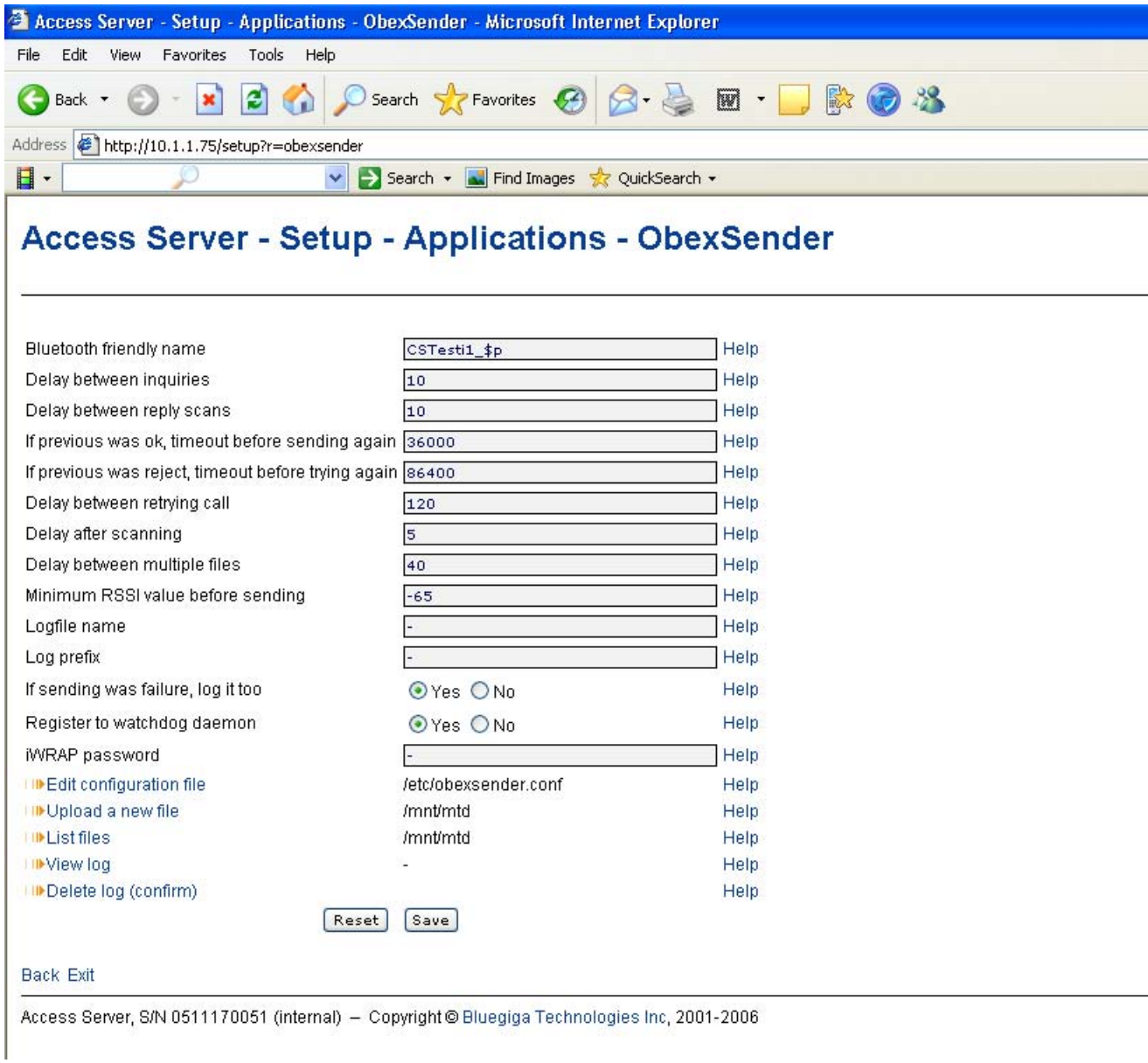

Figure 5-7. Latest Obexsender Main Menu

5.3.3. Ensuring Obexsender is Enabled

By default, the Obexsender application is enabled, so as a first task you should of course enable it if. This is quite simply done from the following page in the WWW setup (Figure 5-8): Access Server - Setup - Applications - Default bootup applications

Obexsender is enabled after a reboot. However, if you have not completed rest of the configuration, do not reboot Access Server yet.

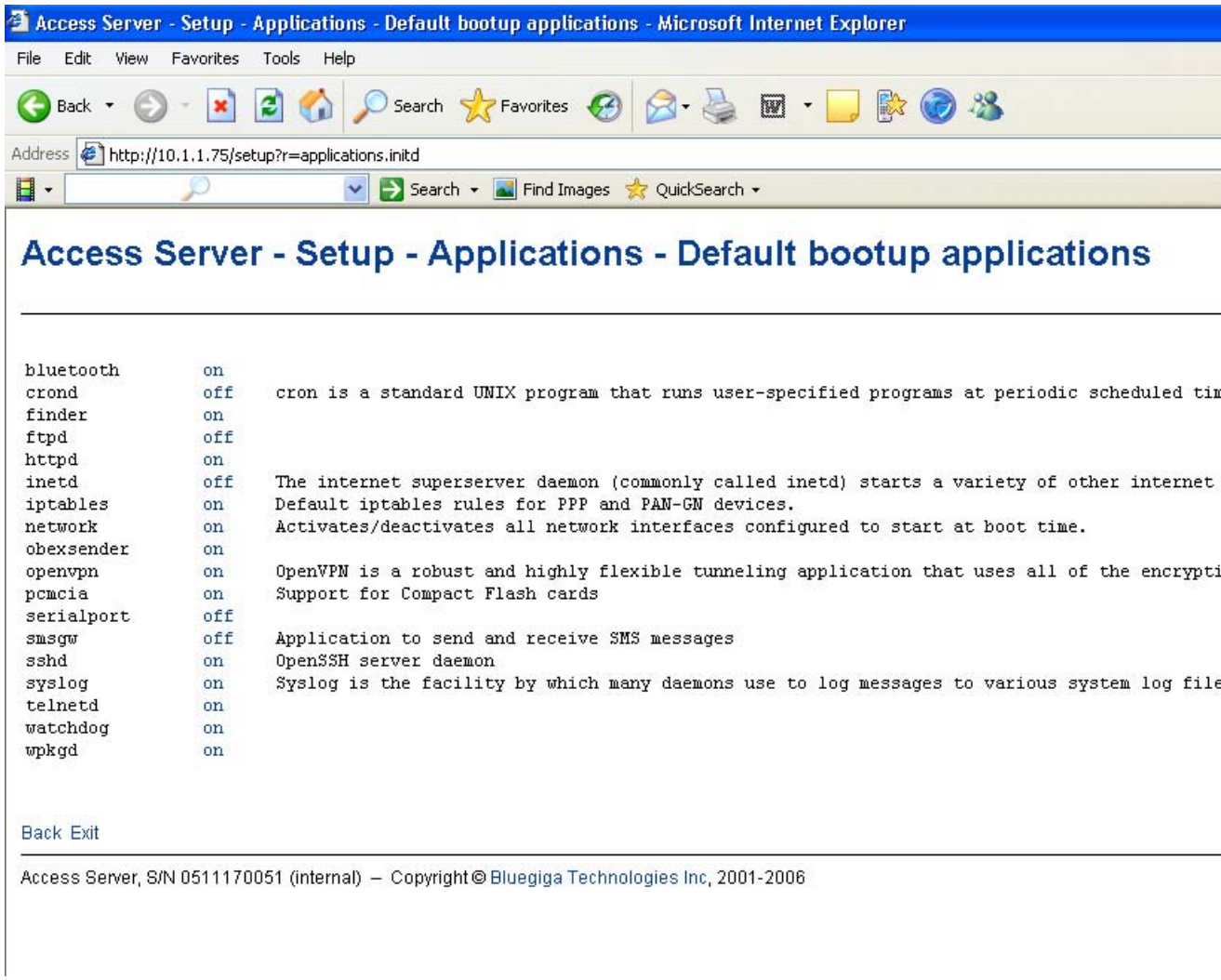

Figure 5-8. Default Boot-up Applications

Note: For Obexsender to start at all, you must define at least one file to be pushed to remote devices. You can do this in:
Access Server - Setup - Applications - ObexSender settings - Edit configuration file
For more information, see Section 5.3.6, chapter "Send these files in this order".

5.3.4. Basic Obexsender Configuration

configuration. As a first step please go the to the WWW setup page in Setup ⟶ Applications ⟶ Obexsender settings.

On this page (Figure 5-7) you can configure the basic Obexsender settings. See Section B.4.3 for default values and detailed descriptions of the settings.

5.3.5. Uploading Files

You can easily upload new content (files) for Obexsender by selecting Upload a new file in the Obexsender main menu. All you need to do is browse for the file you want to upload and

click Upload. You will see a confirmation note, for example *"File /usr/local/obexsender/Bike.jpg uploaded"* .

At the moment, you can only upload to `/usr/local/obexsender` directory using WWW setup. If you would like to upload to another directory, you must use secure FTP to accomplish that. (Normal FTP is disabled by default in Access Server for security reasons). For example WinSCP, available from http://www.winscp.org, is a good application that for secure FTP file transmissions.

5.3.6. Advanced Obexsender Configuration

Specifying the content (files) to be sent by ObexSender is done by editing the `/etc/obexsender.conf` file. The file also contains all configurable ObexSender settings (the settings covered earlier and some advanced settings).

In this section, we will only go through the settings that can not be configured using the WWW interface.

Note: Lines beginning with the hash character "#" are comments and ObexSender will ignore them.

Advanced Configuration Directives

baseband

Specify which iWRAPs are used for sending/inquiry. By default all basebands in this Access Server are in use.

Syntax: `baseband <ip> <port> [password]`

Example:

```
baseband 127.0.0.1 10101
```

ignore

Don't send to these Bluetooth devices. The default setting `ignore 00:07:80:` is recommended. It disables sending files to other Bluegiga Access Servers.

Syntax: `ignore <bdaddr-prefix>`

Example:

```
ignore 00:07:80:
```

tester

Always send to these devices when found (60s interval). Other timeout settings are ignored with these devices.

Syntax: `tester <bdaddr>`

Example:

```
tester 00:07:80:80:00:bf
```

scandir

Obexserver's directory (for remote requests). This is the directory which ObexSender searches for remote requests. It should match the directory configured for Obexserver (`/tmp/obex/` in default configuration.

Syntax: `scandir <directory>`

Example:

```
scandir /tmp/obex
```

file

Specify full pathname(s) of file(s) to be sent, possibly at given time. If there are no files specified, ObexSender does not do inquiry. The files specified are sent in listed order.

Syntax: `file <filename> [timestamp]`

Example for sending `tp1.gif` first, then `tp2.gif`:

```
file /usr/local/obexsender/tp1.gif
file /usr/local/obexsender/tp2.gif
```

Timestamp can be specified as Weekday (Mon/Tue/Wed/Thu/Fri/Sat/Sun), Starthour-Endhour or WeekdayStarthour-Endhour:

Example for sending `image.jpg` on Fridays, `image2.jpg` every day between 8am and 2pm and image3 only on Tuesdays between 8am and 2pm:

```
file /usr/local/obexsender/image1.jpg Fri
file /usr/local/obexsender/image2.jpg 8-14
file /usr/local/obexsender/image2.jpg Tue8-14
```

reply

This feature allows you to request specific content from ObexSender. Basic operation is that you send a file with needed information to Access Server and you will receive a corresponding file in return.

The keyword specified is matched for "<content of file from user> + <bd-ad-dr-es-ss>". Keyword is extended regular expression (regex) and case-non-sensitive.

Syntax: `reply <keyword> <filename>`

Example for replying with `pic.gif` if a GIF image is sent to Access Server (in fact this matches for the string "GIF" found in the image headers; you could use "VCF" for vCards, "JFIF" for JPEG images and so on):

```
reply GIF /usr/local/obexsender/pic.gif
```

Example for replying only to a certain device (its Bluetooth address is already known), ignoring file content (`pic.gif` is sent back after device sends anything to Access Server):

```
reply 00:07:80:80:00:bf /usr/local/obexsender/pic.gif
```

delnomatch

This setting applies if you're using REPLY-feature of ObexSender and you send a file to Access Server to receive specific content. Now, if the file you sent doesn't match to ObexSender configuration, the file will be deleted if this settings is set to "Yes". Otherwise the file is saved. Matching files are always deleted. Disable this if you have some other program doing ObjP/FTP. By default, this is enabled.

Syntax: `delnomatch Yes|No`

Example of disabling the functionality:

```
delnomatch No
```

verbose

Determines the verbosity level of ObexSender logging. The Level can be from 0 to 4, defined by the count of lines with uncommented term `verbose`. Level 0 means that there will be minimal logging and level 4 that there will be maximum amount of logging.

> **Warning**
>
> Full verbose logging (4) should be used only for debugging purposes, since it creates a lot of logs and the flash memory can be filled rather quickly.

Syntax: `verbose`

Example of setting maximum level of ObexSender logging:

```
verbose
verbose
verbose
verbose
```

dumpfile

You can choose to save the information about already served devices, so you can form a so-called "block list". If this block list is saved in flash memory, it will be preserved even if Access Server is rebooted. This basically ensures that remote devices don't receive the same content even if Access Server is rebooted.

Syntax: `dumpfile <filename>`

Example of dumpfile in default location:

```
dumpfile /usr/local/obexsender/ignore.dump
```

dumpdelay

Determines how often (in seconds) a dump file is updated. "0" disables this feature. We recommend to use a rather big value, for example 15min = 900s.

> **Warning**
>
> Using a small value here can physically burn the flash memory over time.

Syntax: `dumpdelay <seconds>`

Example of setting dumpdelay with recommended value:

```
dumpdelay 900
```

broadcast

This settings tells ObexSender to broadcast already served devices to other ObexSenders (specified using unicast IP address, broadcast IP address or interface name).

Syntax: `broadcast <unicast-ip>|<broadcast-ip>|<interface>`

Example of broadcasting to all ObexSender in the same network with the default interface (`nap`):

```
broadcast nap
```

5.3.7. How to Store Files Sent to Access Server

By default, all files sent over Object Push to Access Server are stored to the `/tmp/obex` folder and deleted after they have been processed. It is however possible to save the files to another directory. The following procedure shows how to automatically copy these files to an example folder `/usr/local/remote_request`. (NOTE: you must first create this folder!):

1. Create a copier script `/usr/local/bin/copier`. You can do it for example in the WWW setup -> Advanced settings -> Edit other configuration files and typing here **`/usr/local/bin/copier`**. Put the following script into the file:

```
#!/bin/sh
# to be called from obexsender: --fork /usr/local/bin/copier
# This directory must exist:
SAVEDIR="/usr/local/remote_request"
/bin/cp "$1" "${SAVEDIR}/$3`/bin/date "+%s"`-`echo $1 | /usr/bin/cut -f 2 -d-`"
```

2. Make the script executable by giving command **chmod a+rx /usr/local/bin/copier** at the command line interface.
3. Edit `/etc/bluetooth.conf` and append to the end of the file the following line (below the line is in two parts, combine these in the configuration file):

```
SET BLUETOOTH LISTEN 3 "/usr/sbin/obexserver --bdaddr $b --prefix $b-$P-
--fork /usr/local/bin/copier" 110
```

4. Save changes and restart Access Server.

Now all incoming files are copied to the `/usr/local/remote_request` directory. The format of the files is `bdaddr-btserverport-timestamp-filename.ext`.

5.4. Monitoring Obexsender

Obexsender creates log about its operation to a specified log file. By default, no log file is specified, so you should do this first with instructions provided in Section 5.3.4.

When you choose View log in the Obexsender menu, you can only see the summary of Obexsender action, i.e how many successes, failures and retries have occurred. When you select the date or Total in the summary view, you will see more details. You will see to which Bluetooth address the content was sent and if the transmission was a failure or success, or if transmission will be retried later. See some example logging in the figure below:

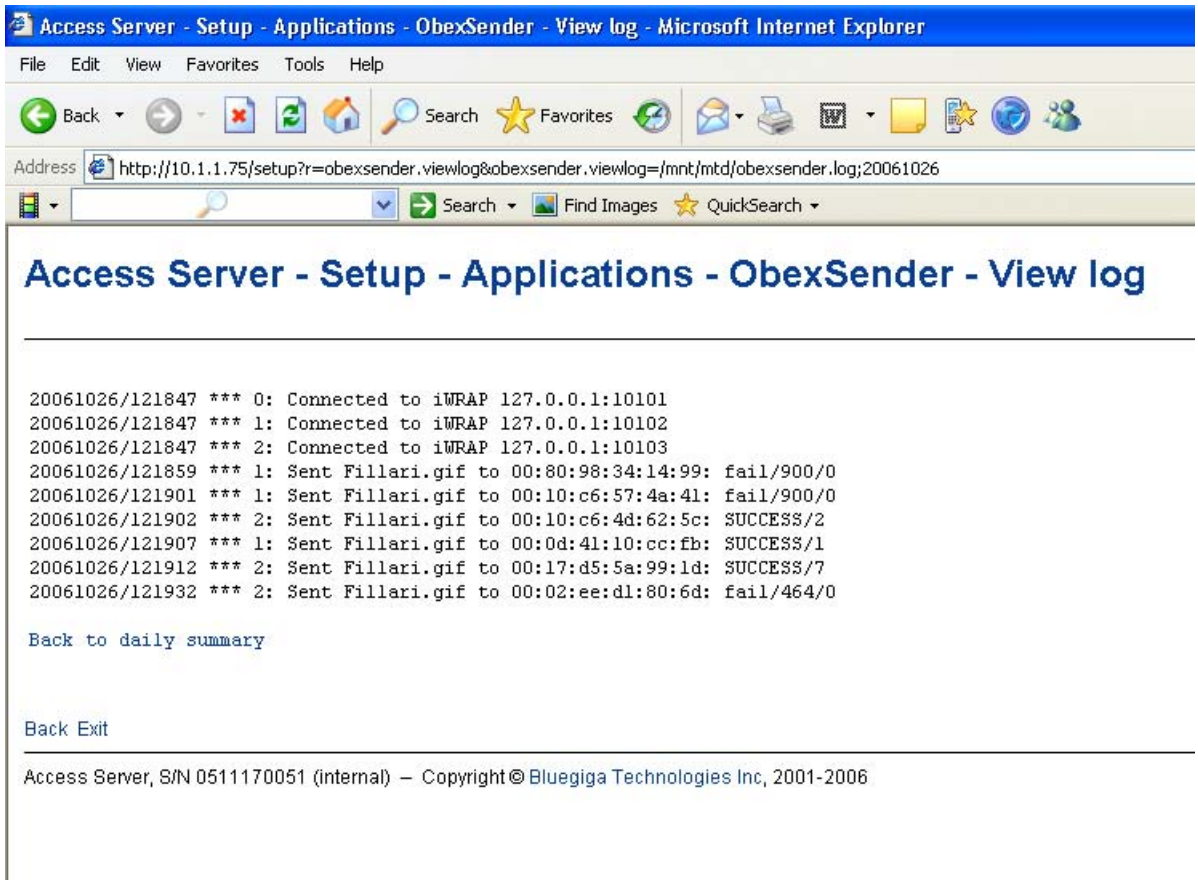

Figure 5-9. Detailed Obexsender Log View

If you want to see even more details about how Obexsender is performing, you can increase the verbosity level of logging. See Section 5.3.6, chapter "Be verbose (0-4)". Full verbose logging is usually needed in problem solving only.

5.5. Troubleshooting and Known Issues

Troubleshooting:

- *Obexsender is not sending anything?*

  Make sure you have at least one content file specified in the configuration file (`obexsender.conf`). See Section 5.3.6, topic "Send these files in this order".

  Also check that Obexsender is activated, see Section 5.3.3.

- *Mobiles receive files only to 10-20 meters. Isn' t Obexsender supposed to work up to 100 meters?*

  Almost all mobile phones are so-called "Class 2" devices, which means that their maximum range is about 10 meters. In good conditions they can achieve even 30 meters.

  If you know there are "Class 1" devices (range up to 100 meters) in the area, you can check the RSSI value you have set, which determines the operational range of Obexsender. See section Section 5.3.4.

Known issues:

- If you enter a non-existing path in "Log file name" configuration, Obexsender will fail to start.
- If you have entered a password for the iWRAP (Bluetooth) interface and the same password is not set in the Obexsender configuration, Obexsender will fail to start.
- If several log files are defined in `obexsender.conf`, Obexsender will fail to start

Chapter 6. Software Development Kit

6.1. Introduction to SDK

This manual describes how to create and use applications by using Access Server's Software Development Environment. The relationships between the applications in the Access Server Software Platform are shown in Figure 6-1.

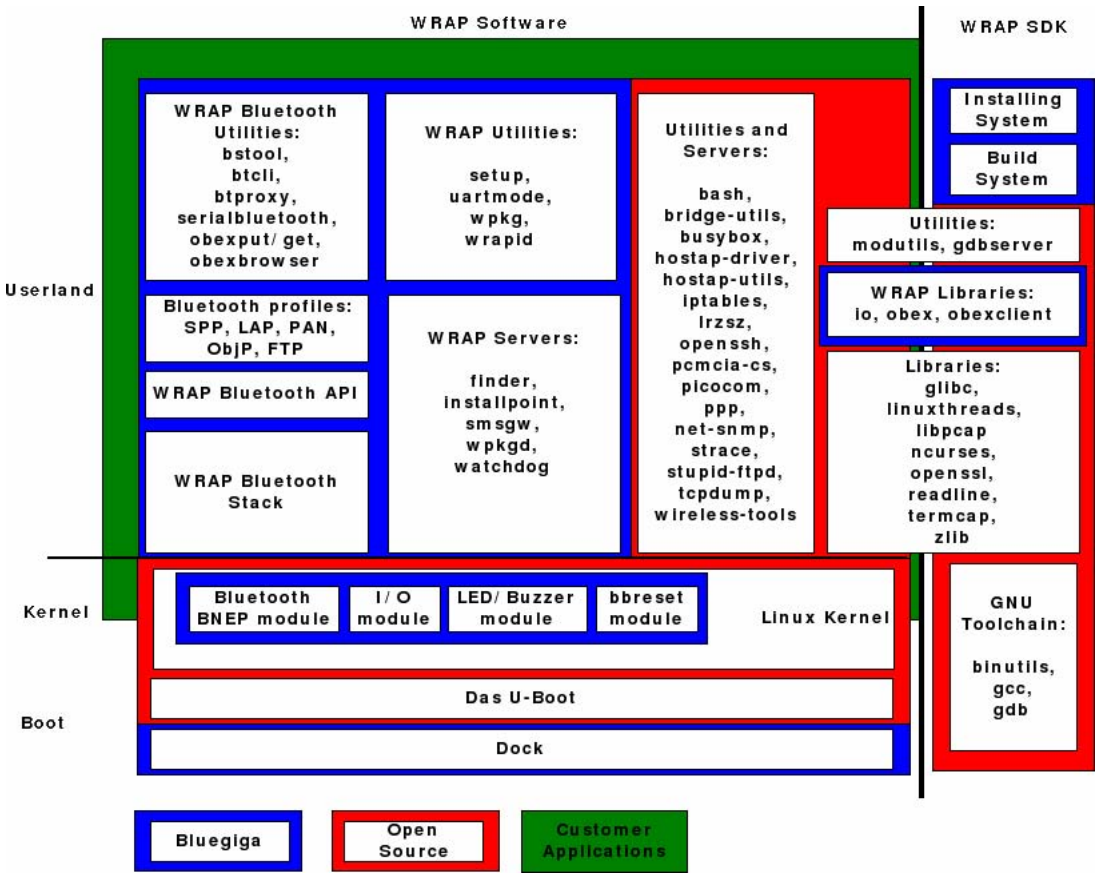

Figure 6-1. Relationship Between Customer Applications and Access Server Software

6.2. Installing SDK

Note: The Software Development Environment can only be installed on a Personal Computer (PC) running the Linux operating system.

6.2.1. Access Server Software Development Environment System Requirements

The following hardware and software are required to run the Access Server Development Environment:

A PC with:

- CD-ROM drive
- The Linux operating system (the SDK has been tested with RedHat Enterprise Linux 3 and above, Fedora Core 2 and above; Suse and Ubuntu are reported to work too)

  `make` and `gawk` must be installed

  Devel libraries (especially `zlib-devel`, `e2fsprogs-devel` and `ncurses-devel`) must be installed

  `modutils-2.4.26` or newer must be installed
- 300MB of available hard disk space

An Ethernet connection to a Local Area Network (also connected to Access Server) is highly recommended.

Mount the Access Server SDK CD-ROM or ISO image, change the current working directory to where it is mounted, and run the **install** script. If the user running **install** does not have privileges to create the directory for the toolchain, normally `/usr/local/arm`, the install script prompts for root's password.

Example (user input is printed **like this**):

```
$ mount /dev/cdrom /mnt/cdrom
$ (or mount -o loop /path/to/sdk2.iso /mnt/cdrom)
$ cd /mnt/cdrom
$ sh install
```

During the installation, the system will prompt you with some questions (described below) regarding the components to install and the paths to install them to. If you are not familiar with Linux, just press **enter** to these questions to accept the default values. The default values are suitable for most users and systems.

6.2.2. Questions Asked by the Install Script

`Access Server toolchain directory` (default: `/usr/local/arm`)

This is the path where you want the Access Server Software Development tools (`arm-linux-gcc`, etc.) to be installed.

> **Note:** If you change this value, the Access Server tools and `libc` must be recompiled. The recompilation process is complicated and lengthy, and it can also fail, depending on your system. Recompilation is automatically done by the install script, if necessary.

`Development directory` (default: `[home_of_current_user]/asdk`)

This is the path where you want the Access Server Software Development Environment to be installed.

`Development directory owner` (default: *`[current_user]`*)

(Asked only if run as root.) This is the development directory owner's username.

Note: If this is not the username of the developer for whom the Software Development Environment is being installed, the user will not have rights to use the development files and therefore can not develop any Access Server software.

`Install toolchain sources` (default: no - unless the tools directory was changed)

This value indicates whether the toolchain sources will be installed. The sources are only required if the Access Server tools directory was changed from the default target location in step 1.

`Compile image after installation` (default: yes)

If set to yes, the install script will compile the Access Server filesystem image to test that the installation was successful and that the Development Environment is working correctly.

6.3. Creating Applications

The fastest way to start developing Access Server applications is to study, change, and recompile the example files in the `asdk/examples` directory.

6.3.1. Application Examples

To demonstrate the software development features of Access Server, the Access Server Software Development Environment comes with several example applications.

6.3.1.1. Installing Examples

The compiled example files are located in WPK packets on the Access Server SDK tree in subdirectories of directory `asdk/examples`.

The examples can be manually uploaded and installed on Access Server by sending them to the `/tmp/obex` directory. The **wpkgd** server automatically installs them. Uploading can be done over Bluetooth, SCP, SFTP or WWW Setup ⟶ Advanced ⟶ Upload a software update (see Figure 2-14).

6.3.1.2. Running Examples

The examples, with their usage and purpose, are described in Table 6-1.

Example	Usage	Purpose
helloworld	**/usr/bin/helloworld**	The "Hello, world!" application.
serial	**/usr/bin/serial /dev/ttyAT1**	"Hello, world!" to the serial port. Notice that `/dev/ttyAT1` must be free (no WRAP SMS Gateway or Bluetooth Serial Port Profile is using it).

Example	Usage	Purpose
forkserver	**SET BLUETOOTH LISTEN 11 /usr/bin/forkserver**	This is the simplest Bluetooth RFCOMM server example. Use, for example, **btserver** as a client to test this example. This example waits for a full line from the client, echoes is back and then exits.
btlogger	**SET BLUETOOTH LISTEN 11 /usr/bin/btlogger /tmp/logfile**	This is a simple Bluetooth RFCOMM server example, which logs lines received from the connected client, and answers with "ACK". Use, for example, **btserver** as a client to test this example.
btserver	**/usr/bin/btserver -** for server mode (if no forkserver is running), **/usr/bin/btserver <bdaddr of btserver in server mode or forkserver> 11** for client mode	This is an advanced iWRAP client example, which can run both as an RFCOMM server, when it works as **forkserver**, or as a client, when it sends "YooHoo" to remote server, waits, displays the response, and quits).
ledtest	**/usr/bin/ledtest**	I/O: LED example.
m2n	**echo testmessage | /usr/bin/m2n**	This is a Machine-2-Network (M2N) example. System Logger (**syslogd**) configuration is needed for actual remote connection. Without it, the example simulates it locally.
www	Browse to http://wrap-ip-address/example.html	Demonstration of the web server capabilities.
makesms	Browse to http://wrap-ip-address/send.html. Notice that this example assumes that WRAP SMS Gateway is up and running (see Section 3.5.3).	This example demonstrates WRAP SMS Gateway by sending SMS messages with it.
setup-helloworld	This example demonstrates how to add a new helloworld submenu to the WWW Setup, with two menu items that change the variables in `/etc/sysconfig/helloworld` file.	

Table 6-1. Examples, Their Usage and Purpose

6.3.2. Creating a New Project

To start a new project, you must create a new subdirectory in your Development Environment's directory (`asdk/`) and add your application source files and `Makefile` to that directory.

A project skeleton can be automatically created by using the Access Server Project AppWizard.

Just give the **make appwiz APP=dir/to/newapp** command in the Development Environment's top level directory (`asdk/`). A "hello world" example ANSI C project is then created.

To use C++ compiler, replace `$(do_link)` with `$(do_link_cc)` in `Makefile`.

The details of the compile process and variables you may need to modify before compiling your application, such as `CFLAGS`, `LDFLAGS` and `CXXFLAGS`, can be seen in file `asdk/Rules.mak`.

Now you have a new project waiting for coding. To compile the project, run **make** in the `asdk/dir/to/newapp` directory.

The build system also creates the installation packet (`hello-timestamp.wpk`), which can be transferred to the `/tmp/obex` directory of Access Server from where it is installed automatically.

6.3.3. Building from the Command Line

The Access Server Development Environment uses the ARM port of the GNU bintools and compilers to build applications. If you are not familiar with Linux development, use the method explained in the previous section instead of writing your own makefiles.

If you still want to use your own development environment, there are two minor issues to remember:

1. Tools are prefixed with **arm-linux-**, so for calling the **gcc** C-compiler, you must call **arm-linux-gcc**, and so on.
2. Tools are located in `/usr/local/arm/3.4.5/bin/` directory, which is not in PATH by default.

6.3.4. Transferring an Application to Access Server

To run an application on Access Server, it must first be transferred to it. There are several ways of doing this (see Section 2.3.3). The most convenient ways in conjunction with software development are discussed in the following subsections.

6.3.4.1. Transferring an Application Using SCP or SFTP

An SCP transfer is done with a single command. In the following example, `myapp` is transferred to the `/tmp` directory in Access Server:

```
$ scp myapp root@<wrap-ip-address>:/tmp
root@<wrap-ip-address>'s password: buffy (not echoed back)
/path/to/myapp/myapp  100%    20KB    20.0KB/s    00:00
$
```

An SFTP transfer is almost similar, but the command procedure resembles an FTP session (FTP can also be used if the FTP server is enabled):

```
$ sftp root@<wrap-ip-address>
Connecting to <wrap-ip-address>...
root@<wrap-ip-address>'s password: buffy (not echoed back)
sftp> cd /tmp
sftp> put myapp
Uploading myapp to /dev/shm/tmp/myapp
```

```
/path/to/myapp/myapp  100%    20KB    20.0KB/s    00:00
sftp> quit
$
```

6.3.4.2. Using SSHFS

With SSHFS, the Access Server filesystem can be securely mounted to be a part of the development host's filesystem.

To download and install SSHFS, visit http://fuse.sourceforge.net/sshfs.html. After installation you can mount the whole filesystem and copy the `myapp` application to the `/tmp` directory in Access Server by using the following commands:

```
$ mkdir mnt
$ sshfs root@<wrap-ip-address>: mnt
root@<wrap-ip-address>'s password: buffy (not echoed back)
$ cp myapp mnt/tmp
$ fusermount -u mnt
$
```

6.3.4.3. Transferring an Application Using Terminal Software

If your Access Server is not connected to a LAN, you can use terminal software of your choice to transfer data to Access Server.

Access Server contains an X/Y/Zmodem protocol application, which allows you to transfer data over the console using almost any terminal software available:

1. Connect your computer to the Access Server management UART using a cross-over serial cable, and start your terminal software (use settings: 115 200bps, 8 data bits, no parity, 1 stop bit).
2. Change your working directory to where you want to upload your application, and run the Xmodem application with your application name as a parameter.
3. Start Xmodem send from your terminal software.

Example 6-1. Transfering Files with Xmodem

```
[root@wrap /] cd /tmp
[root@wrap /tmp] rx testapp
rx: ready to receive testapp.
now start xmodem (checksum, not CRC) send from your terminal
[root@wrap /tmp]
```

If you want to save the application to `/usr/local/bin` (on the flash file system), you will have to replace **cd /tmp** with **cd /usr/local/bin** (and possibly create the directory, if it does not exist). To examine Access Server directory structure, please see Appendix A.

6.3.4.4. Using NFS Mount

To use NFS mount, have a NFS share prepared in your development PC and mount the directory by using command **mount -o nolock <dev-pc-ipaddress>:/nfsshare /mnt/nfs**. After this, you can access the share in directory `/mnt/nfs`.

6.3.5. Running an Application Transferred to Access Server

To run the application you just transferred to Access Server, you need access to the Access Server console, either using terminal software connected to the Access Server management UART or using the SSH connection (log in as user **root** and the root password, which is **buffy** by default).

Having established a connection to Access Server, change the directory to where your application is located and change file permissions so that it can be executed, then run it.

Example 6-2. Running an Application

```
[root@wrap /] cd /tmp
[root@wrap /tmp] chmod 755 testapp
[root@wrap /tmp] ./testapp
```

6.3.6. Using Debugger (GDB/DDD)

You can use GNU debugger GDB and a graphical user interface, such as DDD, for debugging applications in Access Server. This requires that you install **gdbserver** to Access Server. It can be installed from a software package located in directory `asdk/arch/arm/gpl/gdbserver/`

You have to compile with debug options and without symbol stripping to make debugging work. This can be done by overriding the default `CFLAGS` variable set in `asdk/Rules.mak`. You can do this by adding line

```
CFLAGS = -Wall -Os -ggdb -I$(SDKBASE)/include
```

after line

```
include /home/user/asdk/Rules.mak
```

in `Makefile`

After you have compiled your application with these options and transferred your application to Access Server, you can start debugging the application as follows:

1. Start **gdbserver** on Access Server

   Usage:

   `gdbserver :<port> <your application>`

   Example: **gdbserver :6789 ./hello**

2. Start debugger on the host PC. (This example is for the **DDD**)

   Example: **ddd --debugger /usr/local/arm/3.4.5/bin/arm-linux-gdb hello**

3. Create a connection to Access Server.

   Usage:

```
target remote <node IP>:<port>
```

Example: **target remote 192.168.42.3:6789**

4. Run the program by using command **continue**.

6.3.7. Native SDK

It is also possible compile applications for Access Server using native toolchain. To use it, copy files `sdk.iso` and `sdkmount.wpk` from directory `lib` in the Access Server SDK CD-ROM (or ISO image) to the root directory of an USB memory dongle, and insert it to Access Server's USB port. (You can also use Compact Flash memory card for this purpose in similar manner). The native SDK is automatically mounted and you can start using the compiler (**gcc**) in Access Server. All tools now available can be found in directory `/usr/sdk/bin`.

Chapter 7. iWRAP - Bluetooth Interface

The Bluetooth service in Access Server is controlled through the TCP socket interface called iWRAP. The first iWRAP server is listening on port 10101. In the case of Access Server 2293, the second iWRAP server is listening on port 10102, and the third one is listening on port 10103. All commands to an iWRAP server and replies from the server are plain ASCII strings ending in CR+LF ("\r\n"). Commands and replies are not case sensitive.

When connecting to a server, you must first wait for the `READY.` prompt. Do not send any commands prior to this. Some replies are broadcast to all clients of the server. If you see something that you have not requested or that is not intended for your client (identified by the link identifier), simply ignore the reply.

Normally, the iWRAP is protected with the **`buffy`** password. The password can be disabled or changed. For more information, see the SET command. If the password is enabled, it must be sent first, immediately following the `READY.` prompt, to the iWRAP server. Otherwise, all commands will fail.

For an example of using the iWRAP, please see the `asdk/examples/btsend` file in the SDK directory.

In the following examples, **`bold lines`** are commands sent by the client to the iWRAP server and `normal lines` are replies received from the iWRAP server by the client.

7.1. Terms

Bluetooth address (bdaddr) consists of six hex digits separated by a colon. For example, "00:07:80:80:bf:01". With commands requiring a Bluetooth address, you can also use the Bluetooth friendly name instead.

Bluetooth channels are numbered from 1 to 30. In Access Server, the Serial Port Profile is assigned to channel number two, the Object Push Profile and File Transfer Profile to channel number three, and the LAN Access Profile is on channel number four. The other channels are free for user applications.

Link Identifier (link_id) is a number from 0 to 99. It is used to identify established Bluetooth connections.

7.2. Starting the iWRAP Servers

Normally, the iWRAP servers are started automatically upon power-up. You can restart the servers manually (for example, to apply the changes made to the iWRAP settings with the **setup** application without rebooting the system). To restart the servers manually, execute the startup script with option **restart**:

```
[root@wrap /] /etc/init.d/bluetooth restart
```

When the iWRAP servers start up, they use the settings configured with the **setup** application. You can put additional iWRAP commands to the `/etc/bluetooth.conf` file. The commands in that file are processed as the last task every time the iWRAP server is started.

7.3. Writing iWRAP Applications

There are two approaches when writing a iWRAP server program (a program accepting incoming calls) for Access Server, both having different pros and cons:

1. Forklistener
2. iWRAP Client

> **Note:** When writing a client program (that is, a program making an outgoing call), you have to use iWRAP.

7.3.1. Forklistener

This is a standard program reading data from standard input and writing output to standard output. See the SDK directory `examples/forkserver/` for an example of this kind of program.

Pros:

- Easy to write.
- Very robust for simple services.
- You do not have to understand Bluetooth or iWRAP.

Cons:

- Your program is started and stopped for every incoming connection.
- If there are multiple connections, it is not possible to communicate to an external program through one socket.
- You cannot use stdout for debugging; you must use syslog or a log file.
- iWRAP's advanced features are not available: powermodes, MSC, SDP, inquiry, ...

To setup a forklistener, see the **SET** command.

7.3.2. iWRAP Client

iWRAP client is a program communicating with the iWRAP server through control and data sockets. See the SDK directory `examples/btserver/` for an example of this kind of program.

Pros:

- The cons with forklistener do not apply.

Cons:

- More complex than forklistener.
- You must have basic knowledge about Bluetooth and iWRAP.

For documentation about iWRAP, read this chapter carefully.

7.4. Commands Controlling iWRAP

INFO

`INFO` — Get basic info

Synopsis

INFO

Description

INFO is used to retrieve version information on the iWRAP server, in the same format as presented by the `READY.` prompt when the iWRAP connection is opened.

Reply

```
READY. (wrap-2-1-0 $Revision: 1.28 $ bt1.2)
```

QUIT

QUIT — Close iWRAP connection

Synopsis

QUIT

Description

To close the connection to the iWRAP server, use the **QUIT** command.

Reply

There is no reply.

Example

```
READY.
QUIT
```

SET

`SET` — Change parameters

Synopsis

SET [variable [value]]

Description

The **SET** command allows you to alter various Bluetooth and iWRAP parameters. The supported variables are listed in Table 7-1. Issuing a **SET** command without parameters lists the current settings.

Variable	Description
BLUETOOTH BDADDR bdaddr	Our bdaddr. This is a read-only value.
BLUETOOTH NAME friendly_name	You can set your Bluetooth friendly name with this command. Others can request this name with the **NAME** command. You can use the following meta characters: **$S**: Hardware serial number, all ten digits **$s**: Hardware serial number, last three digits **$P**: Server port **$p**: Server port, last digit **$H**: FQDN **$h**: hostname **$$**: $ The default value is **$S_$p**.
BLUETOOTH READABLE mode	If enabled, some SDP result codes will have literal values instead of numeric values. 0: No (always use numeric values) 1: Yes (literal values)
BLUETOOTH CLASS value	You can set the class-of-device value with this command.

Variable	Description
BLUETOOTH LAP value	You can set the IAC LAP value with this command. The default value is 9e8b33
BLUETOOTH ROLE role {policy {timeout}}	You can set the master/slave role switch preference with this command. Optionally, you can also set the link policy and link supervision timeout. The possible values for "role" are: **0**: allow calling, do not request when answering **1**: allow calling, request when answering **2**: do not allow calling, request when answering The default link policy is 000f and the default link supervision timeout is 7d00. See *Bluetooth Specification* for more information on these parameters.
BLUETOOTH ENCRYPT value	This command defines whether to use Bluetooth encryption. To actually enable Bluetooth encryption, the connection must be authenticated. 0: No 1: Yes
BLUETOOTH LAP value	You can set the IAC LAP value with this command. The default value is 9e8b33

Variable	Description
BLUETOOTH PAGEMODE mode {page_timeout {page_repetition_mode {scan_activity_interval scan_activity_window {inquiry_activity_interval inquiry_activity_window}}}}	Pagemode defines whether other devices can find and call you. There are four different modes: 0: No inquiry, no paging 1: Inquiry, no paging 2: No inquiry, paging 3: Inquiry and paging The page timeout is given in hex and the default value is 2000. The default page repetition mode is 2 (R2). The default scan activity is interval 0800 and window 0012 (R1). The default inquiry activity is interval 0800 and window 0012 (R1). See the *Bluetooth Specification* for more information on these parameters.
BLUETOOTH AUTOHIDE physical logical	This command automatically hides the baseband (sets pagemode to 0) if there are more physical ACL links or logical connections than defined. Value 0 means "don't care". Default values: 7 0
BLUETOOTH AUTH * {authflags}	This command removes the default PIN code. If you are making an outgoing connection and the remote end asks for the PIN, "1234" will be sent.
BLUETOOTH AUTH * pin {authflags}	This command sets the default PIN code.
BLUETOOTH AUTH bdaddr {authflags}	This command removes the PIN code for bdaddr.

Variable	Description
BLUETOOTH AUTH bdaddr pin {authflags}	This command sets the PIN code for bdaddr. Authflags are: **--NEWPAIR** Only if we do not have linkkey yet **--REQUEST** Request this PIN from remote, do not reply with this one **--REPLY** Reply to remote requests with this PIN **--CALL** Only if making an outgoing call **--ANSWER** Only when answering to an incoming call **--RFCOMM** Call type is RFCOMM (includes FORK/PPP/...) **--BNEP** Call type is BNEP **--L2CAP** Call type is L2CAP Default authflags are all enabled, except for **--NEWPAIR**. There are three special PINs: **-** Reject without asking PIN. **--** Reject on the connection open, do not check for call types. **+** Accept without asking PIN.
BLUETOOTH PAIR bdaddr linkkey	With this command, you can manually set the linkkey for bdaddr. Note: **SET BLUETOOTH AUTH** must also be set for a value to enable encrypted connections with previously stored link keys.
BLUETOOTH PAIR bdaddr	With this command, you can manually delete the linkkey for bdaddr.
BLUETOOTH PAIREXPIRE seconds	With this command, you can set the expiration time, in seconds, for pairing information.

Variable	Description
BLUETOOTH LISTEN channel cmd {mem {delay}}	This command adds a fork-listener for the channel. When there is an incoming RFCOMM connection to the channel, the iWRAP server handles the connection by itself by forking "cmd". At least "mem" kilobytes of free memory must be available, or the connection will be rejected. After forking, the iWRAP server waits for "delay" timerticks (50ms) before transmitting any data. The client application must modify both the stdout and stdin pipes and set NOECHO, 8BIT and all other necessary modes at the very beginning. The purpose of the "delay" parameter is to give the application enough time to do this.
BLUETOOTH LISTEN channel host:port	This command adds a forward-listener for the channel. When there is an incoming RFCOMM connection to the channel, the iWRAP server will forward it to host:port by using a raw TCP/IP socket.
BLUETOOTH LISTEN psm L2CAP	This command adds an L2CAP listener for the psm.
BLUETOOTH LISTEN channel	This command removes a fork/forward/L2CAP listener from the channel/psm.

Variable	Description
BLUETOOTH LINK mode params	With this command, you can modify the slave's powermode according to the "mode". "params" are optional and mode-dependent. The possible values for "mode" are: 0: Active. Params: None. 1: Park: Round-robin. Params: max_beacon min_beacon sleep_after_unpark sleep_after_round Defaults: 254 160 5 30 Sleeps are specified by timerticks (50ms). 2: Park: Idle. Params: max_beacon min_beacon max_active Defaults: 512 384 6 max_active is the maximum number of active slaves. 3: Sniff: All. Params: max_interval min_interval attempt timeout Defaults: 640 426 1 8 4: Sniff: Idle. Params: idle_timeout max_interval min_interval attempt timeout Defaults: 400 640 426 1 32 idle_timeout is in timerticks (50ms). See *Bluetooth Specification* for more information on params.

Variable	Description
BLUETOOTH QOS service_type token_rate peak_bandwidth latency delay_variation	This command sets default QoS values for a new connection. The parameters are in hex. See *Bluetooth Specification* for more information on params. Defaults: 01 00000000 00000000 000061a8 ffffffff
L2CAP TIMEOUT flushto linkto	With this command, you can define the FlushTimeout and LinkTimeout for L2CAP connections. See *Bluetooth Specification* for more information on params. Defaults: 65535 40000
PPP AUTH	Do not require any PPP authentication on incoming connections.
PPP AUTH username password	Require specified username:password on incoming PPP connections.
PPP CHANNEL channel	Our PPP (LAN Access Profile) channel. The iWRAP server handles this channel internally. If you change this, remember to modify the SDP record as well. Use zero value to disable the LAN Access Profile.
PPP DEFAULTROUTE value	This setting controls whether the iWRAP server should modify the defaultroute setting. There are four different modes: 0: Do no alter defaultroute 1: Set defaultroute according to the outgoing PPP 2: Set defaultroute according to the incoming PPP 3: Set defaultroute according to all PPP calls
PPP WINHANDSHAKE seconds	Timeout to wait for the Windows RAS handshake.
PPP IP ipaddr/mask	This command sets the network IP range for PPP clients.

Variable	Description
PAN ENABLE bitmap	This command controls incoming PAN connections. Bitmap: **1**: Allow incoming PAN-PANU connections. **2**: Allow incoming PAN-GN connections. **4**: Allow incoming PAN-NAP connections. **8**: Enable zeroconf for incoming PAN-PANU connections. **16**: Enable zeroconf for outgoing PAN-PANU connections. The default value "6" is recommended for most cases.
CONTROL AUTOEXEC cmd	Run the **CALL** command, and rerun it when the call is disconnected. Example: **SET CONTROL AUTOEXEC CALL bdaddr PAN-NAP PAN-NAP**
CONTROL PASSWORD	Do not require a password from iWRAP clients.
CONTROL PASSWORD pass {--LOCAL}	Enable password. iWRAP clients must send this password before giving any other command. The password is case sensitive. With an optional **--LOCAL** parameter, clients connecting from localhost are accepted without a password.
CONTROL PING seconds	If this setting is enabled (seconds > 0), the iWRAP server sends a `PING` reply to all iWRAP clients. You have to reply to it with **PONG** or the connection will be closed.
CONTROL WRITETIMEOUT timeticks	With this command, you can set in timeticks (1/20s) how long iWRAP tries to write to the datasocket if it's blocked before giving up and closing the connections.
CONTROL AUTOSAVE what filename	If this setting is enabled, the system automatically saves settings to a file when they change. See the **SAVE** command for possible "what" values.
link_id MSC value	Set MSC for link_id to value. See *ETSI TS 101 369 (GSM 07.10)* for more information.
link_id ACTIVE	With this command, you can set the powermode for a link_id to active.

Variable	Description
link_id PARK params	With this command, you can set the powermode for link_id park. Required "params" are: avg_beacon or max_beacon min_beacon See *Bluetooth Specification* for more information on params.
link_id HOLD params	With this command, you can set the link's powermode to hold. Required "params" are: avg max min See *Bluetooth Specification* for more information on params.
link_id SNIFF params	With this command, you can set the powermode for a link_id to sniff. Required "params" are: avg_interval or max_interval min_interval or max_interval min_interval attempt or max_interval min_interval attempt timeout The default attempt is 1, the default timeout is 8. See *Bluetooth Specification* for more information on params.
link_id QOS service_type token_rate peak_bandwidth latency delay_variation	With this command, you can set the link's QoS values. The parameters are in hex. See *Bluetooth Specification* for more information on params.
link_id MASTER	With this command, you can switch the role to master.
link_id SLAVE	With this command, you can switch the role to slave.

Table 7-1. Supported Parameters for iWRAP SET Command

Reply

When there are parameters, there is no reply.

Example

```
READY.
SET BLUETOOTH NAME Buffy
SET BLUETOOTH PAGEMODE 3
SET BLUETOOTH READABLE 1
SET BLUETOOTH CLASS 020300
SET BLUETOOTH ROLE 0
SET BLUETOOTH ENCRYPT 0
SET BLUETOOTH PAGEMODE 3
SET BLUETOOTH AUTH * 1234
SET BLUETOOTH AUTH 00:07:80:80:bf:01 4242
SET BLUETOOTH AUTH *
SET BLUETOOTH PAIREXPIRE 600
SET BLUETOOTH LISTEN 1 /bin/login 200
SET BLUETOOTH LISTEN 2 "my/own/command with parameters" 100 5
SET BLUETOOTH LISTEN 3
SET PPP DEFAULTROUTE 0
SET PPP AUTH buffy willow
SET PPP AUTH
SET PPP CHANNEL 4
SET PPP WINHANDSHAKE 10
SET PPP IP 192.168.166.0/24

SET 0 MSC 8d

SET CONTROL PING 60
PING
PONG

SET CONTROL PASSWORD

SET CONTROL PASSWORD buffy
<client reconnects>
READY.
SET
ERROR PASSWORD NEEDED.
<client reconnects>
READY.
buffy
SET
SET BLUETOOTH BDADDR 00:07:80:80:bf:01
SET BLUETOOTH NAME Buffy
SET PPP AUTH
SET CONTROL PASSWORD buffy
SET
```

SAVE

SAVE — Save iWRAP settings

Synopsis

SAVE {what} {filename}

Description

The **SAVE** command writes the current settings to a file.

What	Settings
AUTH	SET BLUETOOTH AUTH ...
PAIR	SET BLUETOOTH PAIR ...
BTSET	SET BLUETOOTH ..., but not AUTH or PAIR
OTHERSET	All but SET BLUETOOTH
ALL	Everything

Table 7-1. SAVE parameters

Reply

There is no reply.

Example

```
READY.
SAVE PAIR /etc/bluetooth.pair
SAVE AUTH,PAIR /etc/bluetooth.security
```

LOAD

LOAD — Run iWRAP command script

Synopsis

LOAD {filename}

Description

The **LOAD** command runs commands from a file. This command is usually used with **SAVE** or **SET CONTROL AUTOSAVE** commands.

Reply

There is no reply.

Example

```
READY.
LOAD /etc/bluetooth.security
SET CONTROL AUTOSAVE AUTH,PAIR /etc/bluetooth.security
```

PING

PING — Ask if the connection is alive

Synopsis

PING

Description

The **PING** command can be used to check that the connection to the iWRAP server is alive.

The iWRAP can also send the PING to the client application. In that case, you must reply with the **PONG** command.

Reply

PONG

Example

```
READY.
PING
PONG

PING
PONG
```

PONG

`PONG` — Connection is alive

Synopsis

PONG

Description

The **PONG** command has to be sent back if you see a `PING` reply from the server. If you do not answer, the connection will be closed after a few seconds.

Reply

There is no reply.

Example

```
READY.
PING
PONG
```

ECHO

ECHO — Send a message to other iWRAP clients

Synopsis

ECHO {data}

Description

This command broadcasts its parameters to all iWRAP connections, including the one that sent the command.

Reply

```
ECHO data
```

Example

```
READY.
ECHO Hello world!
ECHO Hello world!
```

LOCK

LOCK — Lock other iWRAP clients

Synopsis

LOCK

Description

This command locks all other iWRAP connections, allowing commands only from this one. This includes all the **PING**s and **PONG**s too. Be polite and do not lock it for a long time.

Reply

There is no reply.

Example

```
READY.
LOCK
UNLOCK
```

UNLOCK

UNLOCK — Unlock other iWRAP clients

Synopsis

UNLOCK

Description

This command opens the lock created by using the **LOCK** command.

Reply

There is no reply.

Example

```
READY.
LOCK
UNLOCK
```

SHUTDOWN

SHUTDOWN — Close iWRAP server

Synopsis

SHUTDOWN

Description

To close the iWRAP server, you can use the **SHUTDOWN** command. This also immediately closes all active connections.

Reply

There is no reply.

Example

```
READY.
SHUTDOWN
```

SLEEP

SLEEP — Wait a second

Synopsis

SLEEP {seconds}

Description

The **SLEEP** command waits for a specified number of seconds before processing further commands.

SLEEP is only usable in rc scripts (/etc/bluetooth.conf).

Reply

There is no reply.

Example

```
READY.
SLEEP 4
```

7.5. Finding Bluetooth Devices

INQUIRY

`INQUIRY` — Search for other devices

Synopsis

INQUIRY [timeout] [NAME] [LAP {lap}]

Description

The **INQUIRY** command is used to search for other Bluetooth devices. The timeout is defined in units of 1.25 seconds. The default timeout is 4 units. If an optional **NAME** parameter is provided, the **NAME** command will be automatically sent to all found devices. The **LAP** option specifies the used IAC LAP; the default value is 9e8b33 (GIAC).

During the inquiry, all devices are listed as soon as they are found by using `INQUIRY_PARTIAL` replies. If the iWRAP server has cached the friendly name of the device found, it is also displayed. When the inquiry times out, a summary is displayed indicating how many devices were found. The summary also repeats the device information.

> **Warning**
>
> Do not use the **NAME** parameter in your program. It is for manual testing only. Use separate **NAME** commands instead.

Reply

```
INQUIRY_PARTIAL bdaddr_of_dev_1 class_of_dev_1 "friendly name" rssi
INQUIRY_PARTIAL bdaddr_of_dev_2 class_of_dev_2 "friendly name" rssi
...
INQUIRY_PARTIAL bdaddr_of_dev_n class_of_dev_n "friendly name" rssi
INQUIRY number_of_devices_found
INQUIRY bdaddr_of_dev_1 class_of_dev_1 "friendly name"
INQUIRY bdaddr_of_dev_2 class_of_dev_2 "friendly name"
...
INQUIRY bdaddr_of_dev_n class_of_dev_n "friendly name"
```

Example

```
READY.
INQUIRY 10
INQUIRY 0

INQUIRY
INQUIRY_PARTIAL 00:07:80:80:bf:01 120300 "willow" 255
INQUIRY_PARTIAL 00:07:80:80:bf:02 520204 "" 255
INQUIRY 2
INQUIRY 00:07:80:80:bf:01 120300 "willow"
INQUIRY 00:07:80:80:bf:02 520204 ""
```

NAME

NAME — Find a friendly name

Synopsis

NAME {bdaddr}

Description

You can ask for the friendly name of another Bluetooth device with the **NAME** command.

Reply

```
NAME bdaddr "friendly name"
NAME ERROR bdaddr reason_code more_info
```

Example

```
READY.
NAME 00:07:80:80:bf:02
NAME 00:07:80:80:bf:02 "buffy"
NAME 00:07:80:80:bf:01
NAME ERROR 00:07:80:80:bf:01 108 HCI_ERR_PAGE_TIMEOUT
```

7.6. Making a Bluetooth Connection

CALL

`CALL` — Connect to other device

Synopsis

CALL {bdaddr} SDP
CALL {bdaddr} {psm} L2CAP
CALL {bdaddr} {channel} RFCOMM
CALL {bdaddr} {uuid} RFCOMM
CALL {bdaddr} {channel} PPP [username password]
CALL {bdaddr} {uuid} PPP [username password]
CALL {bdaddr} {channel} WINPPP [username password]
CALL {bdaddr} {uuid} WINPPP [username password]
CALL {bdaddr} {channel} FORK {"/full/path/to/command and parameters"}
CALL {bdaddr} {uuid} FORK {"/full/path/to/command and parameters"}
CALL {bdaddr} {channel} FORK {host:port}
CALL {bdaddr} {uuid} FORK {host:port}
CALL {bdaddr} {PAN-destUUID} [PAN-srcUUID]

Description

The **CALL** command is used to make a connection to other Bluetooth devices. It returns the link identifier (with an immediate reply), which will be used in subsequent commands and replies.

> **Note:** Always check for a correct link_id before processing replies further.

You can use the special **FORK** call type to create an RFCOMM connection and automatically launch an application, which gets the RFCOMM connection bound to its standard input and output. The client application should modify both the stdout and stdin pipes and set NOECHO, 8BIT and all other necessary modes at the very beginning.

> **Note:** There can only be one pending **CALL** at a time. You have to wait for the `RINGING` event before issuing another CALL. The `RINGING` event comes almost immediately after the **CALL**. You get the `ERROR 008` error if you try to establish another call too quickly. In that case, wait for some tens of milliseconds and retry. Receiving the `CONNECT` or `NO CARRIER` reply may take some time, for example, when the user is keying in the PIN code.
> **Note:** PPP is "raw" PPP without any special handshaking. WINPPP is a Windows RAS handshake followed by raw PPP. If you are unsure, use WINPPP.

Reply

```
CALL link_id
RINGING link_id
```

Example

```
READY.
CALL 00:07:80:80:bf:01 SDP
CALL 0
RINGING 0
CONNECT 0 SDP

CALL 00:07:80:80:bf:01 4 PPP
CALL 1
RINGING 1
CONNECT 1 PPP

CALL NameOfOtherDevice LAN PPP
CALL 1
RINGING 1
CONNECT 1 PPP

CALL 00:07:80:80:bf:02 4 WINPPP buffy willow
CALL 2
RINGING 2
CONNECT 2 PPP

CALL 00:07:80:80:bf:01 1 RFCOMM
CALL 3
RINGING 3
CONNECT 3 RFCOMM 1042

CALL 00:07:80:80:bf:01 2 FORK /bin/login
CALL 4
RINGING 4
CONNECT 4 FORK

CALL 00:07:80:80:bf:01 PAN-NAP
CALL 5
RINGING 5
CONNECT 5 PAN-NAP

CALL 00:07:80:80:bf:02 PAN-NAP PAN-NAP
CALL 6
RINGING 6
CONNECT 6 PAN-NAP

CALL 00:07:80:80:bf:02 2 FORK 127.0.0.1:23
CALL 7
RINGING 7
CONNECT 7 FORK
```

CONNECT

CONNECT — Connected to other device

Synopsis

This is not a command.

Description

CONNECT is not a command, but rather a reply broadcast to you when **CALL** successfully establishes the connection. Remember to check that the link_id matches your **CALL**.

On RFCOMM/L2CAP connections, there is an additional parameter called port. Port refers to the TCP socket port number, which is used to send and receive data to and from the remote device. Connect to the port just like you connected to the iWRAP server. The connection is "raw", which means that no processing of incoming or outgoing data is made.

> **Note:** In the case of L2CAP connections, the data is handled as packets. Therefore, both the incoming and outgoing data must follow the "HDR+L2CAPDATA" format, where HDR is two bytes; first the low byte, and then the high byte of the L2CAPDATA packet length. L2CAPDATA contains the actual L2CAP packet.

Reply

```
CONNECT link_id SDP
CONNECT link_id RFCOMM port
CONNECT link_id L2CAP port
CONNECT link_id PPP
CONNECT link_id FORK
CONNECT link_id PAN-PANU
CONNECT link_id PAN-GN
CONNECT link_id PAN-NAP
```

Example

```
READY.
CALL 00:07:80:80:bf:01 SDP
CALL 0
RINGING 0
CONNECT 0 SDP

CALL 00:07:80:80:bf:01 LAN PPP
CALL 1
RINGING 1
CONNECT 1 PPP

CALL 00:07:80:80:bf:01 1 RFCOMM
CALL 2
RINGING 2
CONNECT 2 RFCOMM 1042
<Client can open socket connection to port 1042>
```

CALL 00:07:80:80:bf:01 2 FORK /bin/login
CALL 3
RINGING 3
CONNECT 3 FORK

CALL 00:07:80:80:bf:01 PAN-NAP
CALL 5
RINGING 5
CONNECT 5 PAN-NAP

CALL 00:07:80:80:bf:02 PAN-NAP PAN-NAP
CALL 6
RINGING 6
CONNECT 6 PAN-NAP

NO CARRIER

`NO CARRIER` — Disconnected from other device

Synopsis

This is not a command.

Description

The `NO CARRIER` reply indicates that you or the remote device closed the active connection, or that your **CALL** failed for some reason.

See Section 7.9 for the list of error codes. Field "more_info" is optional. If present, it gives you a human readable error code or some statistics about the closed connection.

Reply

```
NO CARRIER link_id ERROR reason
NO CARRIER link_id
```

Example

```
READY.
CALL 00:07:80:80:bf:01 4 PPP
CALL 0
RINGING 0
NO CARRIER 0 ERROR 104 HCI_ERR_PAGE_TIMEOUT

CALL 00:07:80:80:bf:01 1 RFCOMM
CALL 1
RINGING 0
CONNECT 1 RFCOMM 1042
NO CARRIER 1 ERROR 000 IN=42,OUT=66,ELAPSED=69
```

RING

RING — Another device is calling you

Synopsis

This is not a command.

Description

The RING reply indicates an incoming call from a remote device. As with CONNECT, on RFCOMM/L2CAP calls there is an additional "port" parameter. Open a socket to the port, if you want to serve this call. PPP and PAN calls are handled internally, which means that you do not have to do anything on them. The iWRAP server closes the connection if nobody grabs the call within 30 seconds.

Special call type REJECTED is used for information only. It is used if somebody tried to call you but was rejected, usually because of failing authentication.

Reply

```
RING link_id bdaddr channel PPP
RING link_id bdaddr channel RFCOMM port
RING link_id bdaddr psm L2CAP port
RING link_id bdaddr PAN-PANU
RING link_id bdaddr PAN-GN
RING link_id bdaddr PAN-NAP
RING link_id bdaddr REJECTED
```

Example

```
READY.
RING 0 00:07:80:80:bf:01 4 PPP

RING 1 00:07:80:80:bf:01 1 RFCOMM 1042
<Client can open socket connection to port 1042>

RING 2 00:07:80:80:bf:01 PAN-GN
```

RINGING

RINGING — Call in progress

Synopsis

This is not a command.

Description

The RINGING reply indicates that a previously initiated outgoing **CALL** is in the state where a new outgoing **CALL** can be made.

Reply

```
RINGING link_id
```

Example

```
READY.
CALL 1 00:07:80:80:bf:01 1 RFCOMM
<Making new CALL is not allowed but generates BUSY error>
CALL 1
<Making new CALL is not allowed but generates BUSY error>
RINGING 1
<Making new CALL is allowed>
CALL 2 00:07:80:80:bf:02 2 RFCOMM
<Making new CALL is not allowed but generates BUSY error>
CALL 2
<Making new CALL is not allowed but generates BUSY error>
RINGING 2
<Making new CALL is allowed>
CONNECT 1 RFCOMM 1042
<Client can open socket connection to port 1042>
CONNECT 2 RFCOMM 1043
<Client can open socket connection to port 1043>
```

CLOSE

CLOSE — Disconnect

Synopsis

CLOSE {link_id}

Description

The **CLOSE** command closes an active connection started with a CONNECT or RING. Note that closing the RFCOMM data socket connection also closes the Bluetooth connection.

Reply

There is no direct reply. NO CARRIER is replied when the connection actually closes.

Example

```
READY.
CALL 00:07:80:80:bf:01 4 PPP
CALL 1
RINGING 1
CONNECT 1 PPP
CLOSE 1
NO CARRIER 1 ERROR 000
```

LIST

`LIST` — List connections

Synopsis

LIST

Description

The **LIST** command reports active connections and some statistics.

Reply

```
LIST number_of_connections
LIST link_id status type blocksize bytes_in bytes_out elapsed_time our_msc
  remote_msc bdaddr channel direction powermode role crypt child_pid hcihandle
LIST link_id status type blocksize bytes_in bytes_out elapsed_time our_msc
  remote_msc bdaddr channel direction powermode role crypt child_pid hcihandle
...
LIST link_id status type blocksize bytes_in bytes_out elapsed_time our_msc
  remote_mscbdaddr channel direction powermode role crypt child_pid hcihandle
```

Reply Values

Status values are:

- WAITING. The iWRAP server is waiting for someone to connect to the datasocket created with the RFCOMM `CONNECT` or `RING` event.
- CONNECTED. The data connection is up and running.
- CLOSING. The datasocket has been closed, and the Bluetooth connection shutdown is in progress.

Type is SDP, RFCOMM, PPP, PAN-PANU, PAN-GN, PAN-NAP, FORK or L2CAP.

Blocksize is the maximum transfer unit of the Bluetooth link; used for statistics only.

Bytes_in and bytes_out refer to the numbers of bytes transferred.

Elapsed_time is the number of seconds the connection has been up.

Msc is the link's MSC value for both ends.

Bdaddr is the Bluetooth address of the connected device.

Channel is the service channel of the connection.

Direction is either OUTGOING or INCOMING.

Powermode is ACTIVE, SNIFF, PARK or HOLD.

Role is MASTER or SLAVE.

Crypt is PLAIN or ENCRYPTED.

Child_pid is the child process ID for types PPP and FORK. The PID is zero for others.

Hcihandle is the HCI handle for this connection.

Example

```
READY.
LIST
LIST 1
LIST 0 CONNECTED RFCOMM 666 4242 100 30 8d 8d 00:07:80:80:bf:01 4
  OUTGOING ACTIVE MASTER PLAIN 0 2a
```

STATUS

`STATUS` — Status of a connection

Synopsis

This is not a command.

Description

The `STATUS` reply is used to inform you about changes in connection status. See also the **SET** command.

Reply

```
STATUS link_id MSC value
```

Example

```
READY.
STATUS 0 MSC 8d
```

7.7. Service Discovery

This section describes the commands used for Bluetooth service discovery and local SDP record manipulation. The commands and their replies use SDP UUID and attribute values, which are listed in the Bluetooth Assigned Numbers documentation. In the commands below, the most useful UUID and attribute values can, however, be replaced with keywords listed in Table 7-3. The same keywords are used in the command replies instead of numeric values, if the parameter **SET BLUETOOTH READABLE** is set to **1**.

Keyword(s)	Value	Hex Value
SDP	UUID_SDP	0001
RFCOMM	UUID_RFCOMM	0003
OBEX	UUID_OBEX	0008
BNEP	UUID_BNEP	000F
L2CAP	UUID_L2CAP	0100
PUBLICBROWSEGROUP, BROWSE, ROOT	UUID_PUBLIC_BROWSE_GROUP	1002
SERIALPORT, SPP	UUID_SERIALPORT	1101
LANACCESS, LAN	UUID_LANACCESS	1102
DIALUPNETWORKING, DUN	UUID_DIALUPNETWORKING	1103
OBEXOBJECTPUSH, OBJP, OPP	UUID_OBEXOBJECTPUSH	1105
OBEXFILETRANSFER, FTP	UUID_OBEXFILETRANSFER	1106
PAN-PANU, PANU	UUID_PANU	1115
PAN-NAP, NAP	UUID_NAP	1116
PAN-GN, GN	UUID_GN	1117
PROTOCOLDESCRIPTORLIST, DESCLIST, DESC	ATTR_PROTOCOLDESCRIPTORLIST	0004
SERVICENAME, NAME	ATTR_SERVICENAME + BASE_LANG_OFFSET	0000 + 0100
SECURITYDESCRIPTION	ATTR_SECURITYDESCRIPTION	030A
NETACCESSTYPE	ATTR_ NETACCESSTYPE	030B
MAXNETACCESSRATE	ATTR_ MAXNETACCESSRATE	030C

Table 7-3. Supported Keywords for Replacing SDP UUIDs or Attributes

SDPSEARCH

`SDPSEARCH` — Browse SDP Records

Synopsis

SDPSEARCH {link_id} {uuid}

Description

The **SDPSEARCH** command is used to send a Service Search Request to a connected SDP server,

identified with link_id. The command only supports searching for one UUID at a time (specified with the uuid parameter, 4 hex digits, or with a keyword), but several requests can be sent during the same SDP connection. However, you must wait for the reply to the previous reply before issuing a new **SDPSEARCH** command.

Reply

```
SDPSEARCH link_id number_of_handles
SDPSEARCH link_id handle_1
SDPSEARCH link_id handle_2
...
SDPSEARCH link_id handle_n
```

Example

```
READY.
CALL 00:07:80:80:bf:01 SDP
CALL 0
RINGING 0
CONNECT 0 SDP
SDPSEARCH 0 LANACCESS
SDPSEARCH 0 1
SDPSEARCH 0 00010000
CLOSE 0
NO CARRIER 0 ERROR 000
```

SDPATTR

SDPATTR — Browse SDP Records

Synopsis

SDPATTR {link_id} {handle} {attribute}

Description

The **SDPATTR** command is used to send a Service Attribute Request to a connected SDP server, identified with the link_id. The command supports requesting for one attribute value (specified with the attribute parameter, 4 hex digits, or a keyword) in one previously retrieved service entry (specified with the handle parameter, 8 hex digits), but several requests can be sent during the same SDP connection. However, you must wait for the reply to the previous reply before issuing a new **SDPATTR** command.

The reply contains the response from the SDP server in encoded form. The code characters are described in Table 7-1.

Char	Description
I	Unsigned integer (2, 4, or 8 hexadecimal digits) follows. This is often a handle, attribute, or attribute value. Attribute values are shown as text if **BLUETOOTH READABLE** is set to **1**.
I	Signed integer byte (2 hexadecimal digits) follows.
U	UUID (4 or 8 hexadecimal digits) follows. Shown as text if **BLUETOOTH READABLE** is set to **1**.
S	String follows.
B	Boolean follows.
<	Start of sequence.
>	End of sequence.
A	Alternative follows.
R	Universal Resource Locator follows.

Table 7-1. SDP Response Formatting Characters

Reply

```
SDPATTR link_id info
```

Example

```
READY.
CALL 00:07:80:80:bf:01 SDP
CALL 0
CONNECT 0 SDP
SDPSEARCH 0 LAN
SDPSEARCH 0 1
SDPSEARCH 0 00010000
SDPATTR 0 00010000 DESCLIST
```

```
SDPATTR 0 < I 0004 < < U 0100 > < U 0003 I 04 > > >
CLOSE 0
NO CARRIER 0 ERROR 000
```

SDPQUERY

SDPQUERY — Browse SDP Records

Synopsis

SDPQUERY {link_id} {uuid} {attribute}

Description

The **SDPQUERY** command is used to send a Service Search Attribute Request to a connected SDP server, identified with the link_id. The command supports requesting for one attribute value (specified with the attribute parameter, 4 hex digits, or a keyword) in all service entries containing one UUID (specified with the uuid parameter, 4 hex digits, or a keyword), but several requests can be sent during the same SDP connection. However, you must wait for the reply to the previous reply before issuing a new **SDPQUERY** command.

Reply

```
SDPQUERY link_id info
```

Example

```
READY.
CALL 00:07:80:80:bf:01 SDP
CALL 0
RINGING 0
CONNECT 0 SDP
SDPQUERY 0 LAN DESCLIST
SDPQUERY 0 < < I 0004 < < U 0100 > < U 0003 I 04 > > > >
SDPQUERY 0 1102 0100
SDPQUERY 0 < < I 0100 S "Lan Access using PPP" > >
CLOSE 0
NO CARRIER 0 ERROR 000
```

SDP bdaddr

`SDP bdaddr` — Check devices SDP

Synopsis

SDP {bdaddr} {uuid}

Description

The **SDP bddaddr** command is the most useful command for retrieving SDP information from the remote device. The command opens the SDP connection, makes the SDP query, closes the connection and replies to the client in encrypted form. The format is described with the **SDPATTR** command.

Reply

```
SDP bdaddr 0 ERROR reason
SDP bdaddr number_of_entries
SDP bdaddr info
SDP bdaddr info
...
SDP bdaddr info
```

Example

```
READY.
SDP 00:07:80:80:bf:01 SERIALPORT
SDP 00:07:80:80:bf:01 1
SDP 00:07:80:80:bf:01 < I SERVICENAME S "Serial Port" >
  < I PROTOCOLDESCRIPTORLIST < < U 0100 > < U RFCOMM I 0b > > >
```

SDP ADD

`SDP ADD` — Add entry to local SDP

Synopsis

SDP ADD {uuid [:uuid2]} {channel} {description}

Description

This command adds a new entry to Access Server's SDP record.

Reply

```
SDP handle
SDP handle ERROR reason
```

Example

```
READY.
SDP ADD LANACCESS 4 "Lan access"
SDP 65536

SDP ADD SERIALPORT 10 "Serial port"
SDP 65537

SDP ADD PAN-NAP 0 "PAN Network Access Point"
SDP 65538

SDP ADD L2CAP:1201 4099 "Private L2CAP for networking"
SDP 65539
```

SDP DEL

`SDP DEL` — Delete entry for local SDP

Synopsis

SDP DEL {handle}

Description

This command deletes one entry from Access Server's SDP record.

Reply

There is no reply.

Example

```
READY.
SDP DEL 65537
```

SDP LIST

`SDP LIST` — List local SDP

Synopsis

SDP LIST

Description

This command lists Access Server's SDP record entries.

Reply

```
SDP number_of_entries
SDP handle uuid channel description
SDP handle uuid channel description
...
SDP handle uuid channel description
```

Example

```
READY.
SDP LIST
SDP 1
SDP 65536 LANACCESS 4 "Lan access"
```

7.8. Example Sessions

Outgoing RFCOMM Call:

```
READY.
CALL 00:07:80:80:bf:01 1 RFCOMM
CALL 2
RINGING 2
CONNECT 2 RFCOMM 1042
STATUS 2 MSC 8d
<Client opens socket connection to port 1042 and transfers data>
CLOSE 2
NO CARRIER 2 ERROR 000
```

Incoming RFCOMM Call:

```
READY.
RING 2 00:07:80:80:bf:01 1 RFCOMM 1042
STATUS 2 MSC 8d
<Client opens socket connection to port 1042 and transfers data>
NO CARRIER 2 ERROR 000
```

7.9. Error Codes

Some commands may reply with an error code. The human-readable name of the error is displayed, if the **SET BLUETOOTH READABLE** setting has value **1**. Error code 8 indicates that the iWRAP server is busy executing a number of commands; there can be several client applications using the stack. Just wait a few seconds and try again. Other error codes indicate unexpected, but often only temporary, communication problems.

You can analyze the error from the numeric code. Values bigger than or equal to 900 are iWRAP errors, described in Table 7-5.

Code	Textual Form	Reason
900	SERVICE_NOT_FOUND	Tried to CALL a device whose SDP records do not include the requested service.
901	ALREADY_CONNECTED	Tried to CALL a device and a service channel that is already connected.
902	OUT_OF_HANDLES	Tried to CALL, but there are too many open connections.
903	INVALID_ADDRESS_<addr>	Tried to CALL a device with a friendly name that could not be found with the inquiry.
904	REJECTED	An incoming call was rejected by the iWRAP server.
905	BUSY	Tried to issue SDPATTR, but another SDP request was in progress.

Code	Textual Form	Reason
906	BUSY	Tried to issue SDPQUERY, but another SDP request was in progress.
907	NOT_CONNECTED	Tried to CLOSE a connection handle that is not active.
908	BUSY	Tried to issue SDPSEARCH, but another SDP request was in progress.
909	INVALID_ADDRESS	Tried to NAME a device with a friendly name that cannot be found with the inquiry.
90a	BUSY	Tried to issue NAME, but another NAME was in progress.

Table 7-5. iWRAP Errors

Other error codes can be analyzed as follows. For example, `NO CARRIER ERROR 465`: The number 465 is hexadecimal, the sum of 0x400 and 0x65, where 0x400 is a mask, which means that this is an RFCOMM level error. 0x65 (decimal 101) means that the RFCOMM error was a connection timeout.

Mask	Error level
0x100	HCI
0x200	L2CAP
0x300	SDP
0x400	RFCOMM

Table 7-6. Errors Masks

The error codes for each mask are listed in the following tables.

HCI Error	Code
HCI_SUCCESS	0
HCI_ERR_UNKNOWN_COMMAND	1
HCI_ERR_NOCONNECTION	2
HCI_ERR_HARDWARE_FAIL	3
HCI_ERR_PAGE_TIMEOUT	4
HCI_ERR_AUTHENTICATION_FAILED	5
HCI_ERR_KEY_MISSING	6
HCI_ERR_MEMORY_FULL	7
HCI_ERR_CONNECTION_TIMEOUT	8
HCI_ERR_MAX_NUM_CONNECTIONS	9
HCI_ERR_MAX_NUM_SCO_CONNECTIONS	10
HCI_ERR_ACL_CONN_ALREADY_EXISTS	11
HCI_ERR_COMMAND_DISALLOWED	12

HCI Error	Code
HCI_ERR_HOST_REJECTED_0D	13
HCI_ERR_HOST_REJECTED_0E	14
HCI_ERR_HOST_REJECTED_0F	15
HCI_ERR_HOST_TIMEOUT	16
HCI_ERR_UNSUPPORTED_PARAM_VALUE	17
HCI_ERR_INVALID_HCI_PARAMETER_VALUE	18
HCI_ERR_OTHER_END_TERMINATE_13	19
HCI_ERR_OTHER_END_TERMINATE_14	20
HCI_ERR_OTHER_END_TERMINATE_15	21
HCI_ERR_CONNECTION_TERMINATE_LOCALLY	22
HCI_ERR_REPEATED_ATTEMPTS	23
HCI_ERR_PARING_NOT_ALLOWED	24
HCI_ERR_UNKNOWN_LMP_PDU	25
HCI_ERR_UNSUPPORTED_REMOTE_FEATURE	26
HCI_ERR_SCO_OFFSET_REJECTED	27
HCI_ERR_SCO_INTERVAL_REJECTED	28
HCI_ERR_SCO_AIR_MODE_REJECTED	29
HCI_ERR_INVALID_LMP_PARAMETERS	30
HCI_ERR_UNSPECIFIED_ERROR	31
HCI_ERR_UNSUPPORTED_LMP_PARAMETER_VAL	32
HCI_ERR_ROLE_CHANGE_NOT_ALLOWED	33
HCI_ERR_LMP_RESPONSE_TIMEOUT	34
HCI_ERR_LMP_ERROR_TRANSACTION_COLLISION	35
HCI_ERR_LMP_PDU_NOT_ALLOWED	36
HCI_ERR_ENCRYPTION_MODE_NOT_ACCEPTABLE	37
HCI_ERR_UNIT_KEY_USED	38
HCI_ERR_QOS_NOT_SUPPORTED	39
HCI_ERR_INSTANT_PASSED	40
HCI_ERR_PAIRING_WITH_UNIT_KEY_NOT_SUPP	41
HCI_ERR_ILLEGAL_HANDLE	100
HCI_ERR_TIMEOUT	101
HCI_ERR_OUTOFSYNC	102
HCI_ERR_NO_DESCRIPTOR	103

Table 7-7. HCI Error Codes

L2CAP Error	Code
L2CAP_NO_CAUSE	0
L2CAP_ERR_PENDING	1
L2CAP_ERR_REFUS_INV_PSM	2

L2CAP Error	Code
L2CAP_ERR_REFUS_SEC_BLOCK	3
L2CAP_ERR_REFUS_NO_RESOURCE	4
L2CAP_ERR_TIMEOUT_EXTERNAL	0xee

Table 7-8. L2CAP Error Codes

SDP Error	Code
SDP_ERR_RESERVED	0
SDP_ERR_UNSUPPORTED_SDP_VERSION	1
SDP_INVALID_SERVICE_RECORD_HANDLE	2
SDP_INVALID_REQUEST_SYNTAX	3
SDP_INVALID_PDU_SIZE	4
SDP_INVALID_CONTINUATION_STATE	5
SDP_INSUFFICIENT_RESOURCES	6
SDP_ERR_UNHANDLED_CODE	100
SDP_ERR_TIMEOUT	101
SDP_ERR_NOTFOUND	102
SDP_INVALID_RESPONSE_SYNTAX	103
SDP_NOT_FOUND (not really an error)	200

Table 7-9. SDP Error Codes

RFCOMM Error	Code
RFCOMM_SUCCESS	0
RFCOMM_ERR_NORESOURCES	1
RFCOMM_ERR_ILL_PARAMETER	2
RFCOMM_ERR_REJECTED (Connection setup was rejected by remote side)	100
RFCOMM_ERR_TIMEOUT (Connection timed out)	101
RFCOMM_ERR_NSC (Non supported command received)	102
RFCOMM_ERR_ILLPARAMETER	103

Table 7-10. RFCOMM Error Codes

If the problems persist after restarting the communication parties, please contact Bluegiga Technologies as instructed in Section 1.2.

Chapter 8. I/O API

The Bluegiga I/O API defines how to access Access Server's LEDs, buzzer, and general purpose I/O.

8.1. Led and Buzzer API

Access Server's LEDs and buzzer can be accessed through the `/dev/led` device. You can check the status of the LEDs and the buzzer with the **cat /dev/led** command and set LEDs or the buzzer with the **echo abcde > /dev/led** command. An upper case letter means that the LED or buzzer is ON, a lower case letter means that the LED or buzzer is OFF. Letter "a" is the buzzer, letters "b".."e" are LEDs 1..4.

Example:

```
[root@wrap /] echo abCDe > /dev/led
```

8.2. GPIO API

The Digital I/O pins of Access Server can be controlled write-only by using the `/dev/io` device in the same way as the `/dev/led` device for LEDs and buzzer described above.

The letter-to-I/O mapping of the 16 pins is as follows, when looking at the connector:

```
hgfedcba
Xijklmno
```

X is the ground pin (and cannot be set).

o is the voltage sense pin (user can use any voltage from 3.3V to 5.0V).

The I/O must first be enabled by using the **echo Z > /dev/io** command. After that, pins can be driven up by echoing the corresponding upper case letter (A-N) or down by echoing a lower case letter (a-n) to the `/dev/io` device.

Example:

```
[root@wrap /] echo ZaBcD > /dev/io
```

Chapter 9. Advanced Use Cases for Access Server

This chapter will give you advanced use cases for Access Server. The cases listed here are not so trivial, the simple cases are already listed mostly in Chapter 7.

9.1. Making Access Server Secure

TBA

9.2. Saving Bluetooth Pairing Information Permanently

By default, Access Server discards pairing information after 30 minutes and does not store pairing data permanently. Therefore, rebooting of Access Server removes all pairing information.

To increase the pairing data timeout and to automatically store the pairing data to the permanent storage and to automatically reload the information at reboot, append the following iWRAP commands to the end of `/etc/bluetooth.conf` file (Setup ⟶ Bluetooth settings ⟶ Edit startup script in WWW Setup):

```
# Set pairing data timeout to ~370 days (in seconds)
# Note: timeout counter is restarted at reboot
SET BLUETOOTH PAIREXPIRE 32000000

# Automatically load the pairing data
LOAD /etc/bluetooth.security$p

# Automatically save the pairing data
SET CONTROL AUTOSAVE AUTH,PAIR /etc/bluetooth.security$p
```

Note: Do not forget *$p* from the filename. It is replaced with the Bluetooth baseband number. On a multiradio Access Server, forgetting it will make the security data to be overwritten by the other Bluetooth processes.
Note: Pairing must be done between each Bluetooth device pairs. There is no way of making a single pairing between a device and all three basebands of the WRAP 2293 Access Server.

9.3. Digital Pen

Access Server will support most of the digital pens. The examples below are for Nokia Digital Pen SU-1B but they should apply to other pens too.

To setup Access Server for digital pens you have to give following iWRAP commands. The best way to do this is to append the following line to `/etc/bluetooth.conf` file (Setup ⟶ Bluetooth settings ⟶ Edit startup script in WWW Setup):

```
# Load Digital Pen emulation commands
LOAD /etc/bluetooth.pen
```

The `/etc/bluetooth.pen` must then be created (in WWW Setup, you can do it at Setup ⟶ Advanced settings ⟶ Edit other configuration files). It should contain the lines following the example below:

```
# Emulate a phone
SET BLUETOOTH CLASS 500204
```

```
SET BLUETOOTH LISTEN 1 "*/usr/sbin/dun"
SDP ADD DUN 1 "Digital Pen DUN"

# Add two pens and their pin codes
SET BLUETOOTH AUTH 00:07:cf:51:f6:8e 9079 --REPLY
SET BLUETOOTH AUTH 00:07:cf:51:d5:2b 6603 --REPLY
# Note: See pen's manual for correct bluetooth address and pin code

# Optionally reject all other incoming connections
SET BLUETOOTH AUTH * - --NEWPAIR
```

After these settings you can pair and use the digital pen with Access Server just like you would use it with a phone. Both modes, receiving pictures to Access Server, and external server via dialup, are supported.

9.4. OpenVPN

This chapter explains how to create a secure network between your Access Server and a PC running Windows OS. This is done using Virtual Private Networking (VPN) and the particular software in use is OpenVPN, which is open source software and is available for everyone without charge. VPN creates a secure tunnel between Access Server and a PC, which enables you, for example, to control a GPRS connected Access Server in a remote location.

9.4.1. Prerequisites

First, download OpenVPN from http://openvpn.se. A normal OpenVPN version using plain command line interface is available in http://openvpn.net/download.html. The basic instructions naturally apply for both versions, since the actual software is the same. OpenVPN GUI is only available for Windows OS.

For Access Server, you must download the OpenVPN installation packet from www.bluegiga.com/techforum. If you do not have access to the Tech forum, you can apply for access in the same site. In the Tech forum, go to Access Server -> Downloads, where you can find the installation packet called `openvpn-2.0.8-1.wpk`. Access Server is a Linux system, and only command line interface is provided at this point.

This guide relies on material provided in http://openvpn.net/. If you want more specific information on features described here or other features OpenVPN provides, please visit http://openvpn.net/howto.html.

9.4.2. Installing OpenVPN

In Windows, execute the installation file and wait until it is complete. There should be no need for reboot. After this, the OpenVPN icon appears in the system tray. Right-click the icon and you can see the available options

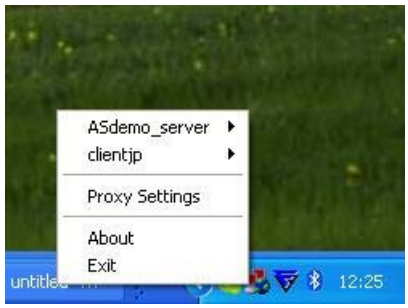

Figure 9-1. OpenVPN GUI Options Menu

In Access Server, the easiest way to install OpenVPN is through the WWW setup. Just enter the server IP address in you web browser and log in. If you do not know the IP address, you can use the WRAPfinder application to find out the IP address. WRAPfinder is located in the CD provided with the server.

When in WWW setup, go to Advanced settings -> Upload a software update. There you can choose the `openvpn-2.0.8-1.wpk` installation packet and upload it to the server. After this you can go back to the Advanced settings page and choose List installed software components. If you can see openvpn in this list, the installation is complete.

9.4.3. Creating Certificates and Keys

In this chapter, we create the necessary files to ensure privacy in the VPN, i.e. we will establish a Public Key Infrastructure (PKI). The PKI consists of:

- A master Certificate Authority (CA) certificate and key which is used to sign each of the server and client certificates.
- A separate certificate (also known as a public key) and private key for the server and each client.

OpenVPN uses bi-directional authentication, which means that both server and client will authenticate each other using certificates before connection is considered safe.

To create the files we will use a set of scripts bundled with OpenVPN for Windows. To see how the same thing is done in Linux, see http://openvpn.net/howto.html#pki.

In Windows, open up a Command Prompt window and go to `\Program Files\OpenVPN\easy-rsa`. Run the following batch file to copy configuration files into place (this will overwrite any existing `vars.bat` and `openssl.cnf` files):

```
init-config
```

Now, edit the `vars` file (called `vars.bat` on Windows) and set the *`KEY_COUNTRY`*, *`KEY_PROVINCE`*, *`KEY_CITY`*, *`KEY_ORG`*, and *`KEY_EMAIL`* parameters. Do not leave any of these parameters blank.

```
vars
clean-all
build-ca
```

The **build-ca** builds the certificate authority (CA) certificate and key by invoking the interactive **openssl** command:

```
ai:easy-rsa # ./build-ca
Generating a 1024 bit RSA private key
............++++++
............++++++
writing new private key to 'ca.key'
-----
You are about to be asked to enter information that will be incorporated
into your certificate request.
What you are about to enter is what is called a Distinguished Name or a DN.
There are quite a few fields but you can leave some blank
For some fields there will be a default value,
If you enter '.', the field will be left blank.
-----
Country Name (2 letter code) [FI]:
State or Province Name (full name) [NA]:
Locality Name (eg, city) [ESPOO]:
Organization Name (eg, company) [OpenVPN-TEST]:
Organizational Unit Name (eg, section) []:
Common Name (eg, your name or your server's hostname) []:OpenVPN-CA
Email Address [me@myhost.mydomain]:
```

> **Note:** In the above sequence, the most queried parameters were defaulted to the values set in the `vars` or `vars.bat` files. The only parameter which must be explicitly entered is the *`Common Name`*. In the example above, we have used "OpenVPN-CA".

Next, we will generate a certificate and private key for the server:

```
build-key-server server
```

As in the previous step, most parameters can be defaulted. When the *`Common Name`* is queried, enter "server". Two other queries require positive responses, "Sign the certificate? [y/n]" and "1 out of 1 certificate requests certified, commit? [y/n]".

Generating client certificates is very similar to the previous step:

```
build-key client
```

If you want to use many clients, then you could use, for example, the following commands:

```
build-key client1
build-key client2
build-key client3
```

In this case, remember that for each client, make sure to type the appropriate *`Common Name`* when prompted, i.e. "client1", "client2", or "client3". Always use a unique common name for each client.

Next we'll create Diffie Hellman parameters that must be generated for the OpenVPN server:

```
build-dh
```

The output is as follows:

```
ai:easy-rsa # ./build-dh
Generating DH parameters, 1024 bit long safe prime, generator 2
This is going to take a long time
.................+.......................................
...................+.............+..................+.........
.....................................
```

Now you can find the generated keys and certificates in the `keys` subdirectory. The final step in the key generation process is to copy all files to the machines which need them, taking care to copy secret files (`server.key` and `client.key`) over a secure channel.

9.4.4. Creating Configuration Files

Both the server and client devices must have certain configuration files for OpenVPN to determine, for example, which IP addresses to use. In this chapter, we will create a basic configuration file for OpenVPN server and client. We'll make the PC as server and Access Server as the client. An example configuration files can be found here: http://openvpn.net/howto.html#examples. In our example, we use most of the setting described in these files.

> **Note:** The configuration files can be named, for example, `server.conf` and `client.conf` in a Linux system. On Windows they would be named `server.ovpn` and `client.ovpn`, where the file extension is different.

9.4.4.1. Server Configuration File

There are lots of configuration options that can be used with OpenVPN, but this guide only covers the basic approach to set up a working VPN with minimal effort. The lines needed in the server configuration file are listed below. After each line, an explanation follows, see Figure 9-2:

```
port 1194
```

- Determines the TCP or UDP port that OpenVPN should listen to. For multiple OpenVPN instances on the same machine, you'll need to use a different port for each one. Make sure your firewall allows traffic through these ports.

```
proto udp
```

- Determines whether to use TCP or UDP. We have chosen UDP in our application.

```
dev tun
```

- Determines whether to use routed IP channel (tun) or an Ethernet tunnel, i.e. Ethernet bridging (tap). 'tap' creates a virtual Ethernet adapter, while 'tun' device is a virtual point-to-point IP link. We have chosen 'tun' because of its better efficiency and scalability.

```
ca "C:\\Program Files\\OpenVPN\\config\\ca.crt"
```

- This is a so-called master Certificate Authority (CA) certificate. This will be placed in both the server and client devices, it's the same for all devices. Since the server is a Windows machine, we need to use double backslashes (\\ ) in pathnames. In Linux system one slash (/) is used.

```
cert "C:\\Program Files\\OpenVPN\\config\\server.crt"
```

- This is the certificate (a.k.a public key) for the server device.

```
key "C:\\Program Files\\OpenVPN\\config\\server.key"
```

- This is the private key for the server device and it should be kept secret.

```
dh "C:\\Program Files\\OpenVPN\\config\\dh1024.pem"
```

- This file refers to Diffie-Hellman key exchange, which is a cryptographic protocol that allows two devices that have no prior knowledge of each other to establish a shared secret key over an insecure connection.

```
server 172.30.203.0 255.255.255.0
```

- Here we create the VPN subnet. In this example, the server will take 172.30.203.1 for itself, the rest will be left for clients to use. Each client will be able to reach the server on 172.30.203.1.

```
ifconfig-pool-persist C:\\Program Files\\OpenVPN\\config\\Logs\\ipp.txt
```

- This file maintains a record of client <-> virtual IP address associations. If OpenVPN goes down or is restarted, reconnecting clients can be assigned the same virtual IP address that was previously assigned.

```
keepalive 10 120
```

- This feature causes ping-like messages to be sent back and forth over the link so that each side knows when the other side has gone down. The default parameter "10 120" makes ping occur every 10 seconds and remote peer is assumed down if no ping is received within 120 seconds.

```
persist-key
```

- Persist features try to avoid accessing certain resources on restart that may no longer be accessible.

```
persist-tun
```

- See above.

```
status C:\\Program Files\\OpenVPN\\config\\Logs\\openvpn-status.log
```

- OpenVPN outputs a short status description to this file showing current connections. This file is truncated and rewritten every minute.

```
verb 3
```

- This sets the verbosity level of the log file.
  - 0 is silent, except for fatal errors
  - 4 is reasonable for general use
  - 5 and 6 can help to debug connection problems
  - 9 is extremely verbose

```
tls-timeout 4
```

- Packet retransmit timeout on TLS control channel if no acknowledgment from remote end within n seconds (n = 4 in this example).

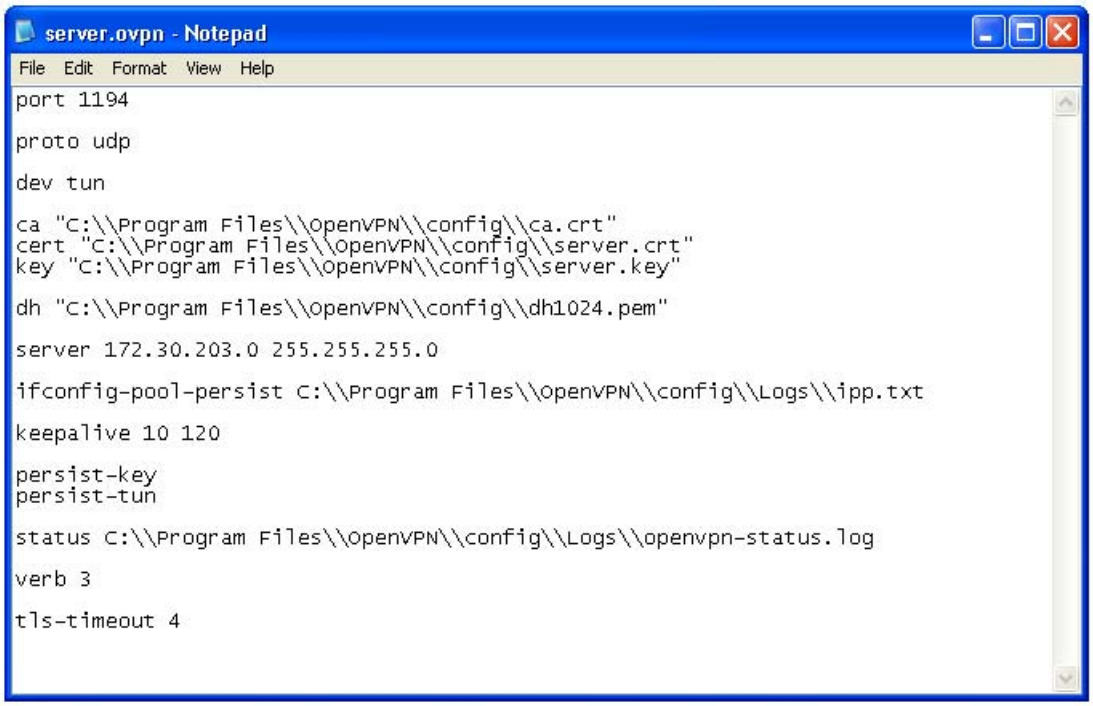

Figure 9-2. Server Configuration File

9.4.4.2. Client Configuration File

Just like with the server configuration file, we'll describe here the basic client settings needed in our example setup, see Figure 9-3:

```
client
```

- Here we specify that we are a client and that we will be pulling certain config file directives from the server.

```
dev tun
```

- This setting is the same as in the server configuration file. Use the same setting you're using in the server.

```
proto udp
```

- This setting is the same as in the server configuration file. Use the same setting you're using in the server.

```
remote 10.1.1.35 1194
```

- This setting configures the hostname/IP and port of the server.

```
resolv-retry infinite
```

- Keep trying indefinitely to resolve the host name of the OpenVPN server. Very useful on machines which are not permanently connected to the internet, such as laptops.

```
nobind
```

- Most clients don't need to bind to a specific local port number.

```
persist-key
```

- This setting is the same as in the server configuration file. Use the same setting you're using in the server.

```
persist-tun
```

- This setting is the same as in the server configuration file. Use the same setting you're using in the server.

```
ca /usr/local/openvpn/conf/ca.crt
```

- This is the same `ca.crt` file as in the server. See server config file descriptions for more information.

```
cert /usr/local/openvpn/conf/client.crt
```

- This is the certificate (a.k.a public key) for the client device.

```
key /usr/local/openvpn/conf/client.key
```

- This is the private key for the client device.

```
verb 3
```

- Sets the verbosity level of the log file.

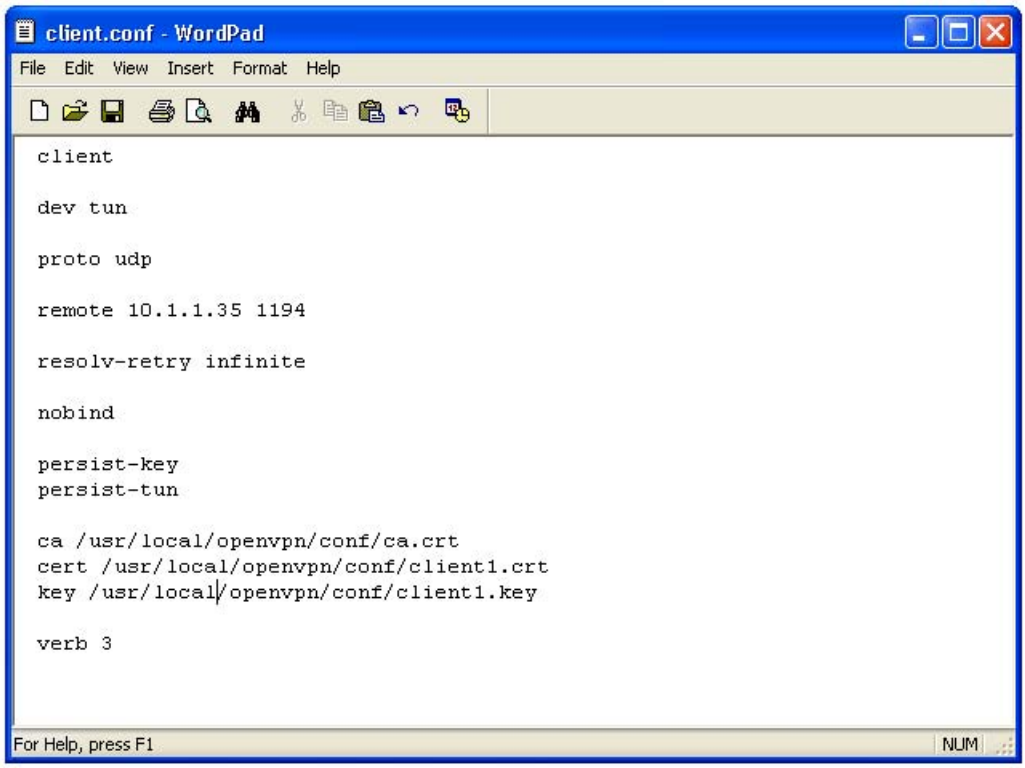

Figure 9-3. Client Configuration File

9.4.5. Starting up VPN

First, place the configuration files in the client and server. Like in the examples, the location for these files can be, for example, `C:\Program Files\OpenVPN\config` in Windows and `/usr/local/openvpn/config` in Linux. Next, copy the authentication files (`ca.crt`, `server.crt`, `server.key`, `client.crt` and `client.key`) into the same directories.

9.4.5.1. Starting up the Server

The OpenVPN server must be accessible from the internet:

- open UDP port 1194 on the firewall (or the TCP/UDP port you've configured), or
- set up a port forward rule to forward UDP port 1194 from the firewall/gateway to the machine running the OpenVPN server
- make sure TUN/TAP device is allowed access through firewalls

To start the OpenVPN server right-click on the `.ovpn` file on Windows and choose "Start OpenVPN on this config file" or by right-clicking the GUI icon on taskbar and start correct config file from there. It's also possible to start from command line:

```
openvpn [server_config_file]
```

Where "server_config_file" is in our Windows examples is `server.ovpn`.

A normal server startup should look like this (output will vary across platforms):

```
Sun Feb  6 20:46:38 2005 OpenVPN 2.0_rc12 i686-suse-linux [SSL] [LZO] [EPOLL] built on Feb
Sun Feb  6 20:46:38 2005 Diffie-Hellman initialized with 1024 bit key
Sun Feb  6 20:46:38 2005 TLS-Auth MTU parms [ L:1542 D:138 EF:38 EB:0 ET:0 EL:0 ]
Sun Feb  6 20:46:38 2005 TUN/TAP device tun1 opened
Sun Feb  6 20:46:38 2005 /sbin/ifconfig tun1 10.8.0.1 pointopoint 10.8.0.2 mtu 1500
Sun Feb  6 20:46:38 2005 /sbin/route add -net 10.8.0.0 netmask 255.255.255.0 gw 10.8.0.2
Sun Feb  6 20:46:38 2005 Data Channel MTU parms [ L:1542 D:1450 EF:42 EB:23 ET:0 EL:0 AF:3/
Sun Feb  6 20:46:38 2005 UDPv4 link local (bound): [undef]:1194
Sun Feb  6 20:46:38 2005 UDPv4 link remote: [undef]
Sun Feb  6 20:46:38 2005 MULTI: multi_init called, r=256 v=256
Sun Feb  6 20:46:38 2005 IFCONFIG POOL: base=10.8.0.4 size=62
Sun Feb  6 20:46:38 2005 IFCONFIG POOL LIST
Sun Feb  6 20:46:38 2005 Initialization Sequence Completed
```

9.4.5.2. Starting up the Client

We'll start the client from Linux command line:

```
openvpn [client_config_file]
```

Where "client_config_file" is in our examples `client.conf`.

A normal client startup looks similar to the server output and should end with the "Initialization Sequence Completed" message.

Now, try a ping across the VPN from the client:

```
ping 10.8.0.1
```

If the ping succeeds, you have a functioning VPN.

Chapter 10. Certification Information and WEEE Compliance

Access Server is CE approved and Bluetooth qualified v. 2.0 + EDR. It has been measured against the following specification standards: ETSI EN 300 328 v1.6.1 / EN 301 489-1/17 / EN 60950-1 / FCC parts 15.247, 15.209, 15.207, 15.109 and 15.107. Supported Bluetooth profiles are: GAP, SDAP, LAN client and server, SPP A and B, FTP client and server, ObjP client and server, PAN-PANU, PAN-GN and PAN-NAP.

Hereby, Bluegiga Technologies declares that this Access Server is in compliance with the essential requirements and other relevant provisions of Directive 1999/5/EC.

This device complies with Part 15 of the FCC Rules.

The device operation is subject to the following two conditions:

1. This device may not cause harmful interference, and
2. This device must accept any interference received, including interference that may cause undesired operation.

This equipment has been tested and found to comply with the limits for a Class B digital device, pursuant to Part 15 of the FCC Rules. These limits are designed to provide reasonable protection against harmful interference in a residential installation. This equipment generates, uses, and can radiate radio frequency energy and, if not installed and used in accordance with the instructions, may cause harmful interference to radio communications. However, there is no guarantee that interference will not occur in a particular installation.

If this equipment does cause harmful interference to radio or television reception, which can be determined by turning the equipment off and on, the user is encouraged to try to correct the interference by one or more of the following measures:

- Reorient or relocate the receiving antenna
- Increase the distance between the equipment and receiver
- Connect the equipment into an outlet on a circuit different from that to which the receiver is connected
- Consult the dealer or an experienced radio or television technician for help

Warning

Changes or modifications made to this equipment not expressly approved by Bluegiga Technologies Inc. may void the FCC authorization to operate this equipment.

The radiated output power of Access Server is far below the FCC radio frequency exposure limits. Nevertheless, Access Server should be used in such a manner that the potential for human contact during normal operation is minimized.

To meet the FCC's exposure rules and regulations:

- The antenna(s) used for this transmitter must be installed to provide a separation distance of at least 20 cm from all the persons.

- Any transmitter installed in the CF card slot must not exceed 4 W of e.i.r.p. To check if a particular equipment complies with this restriction, you need to know its FCC ID number and visit the searching engine in the FCC web site in the following Internet address, where you can find the output power by the equipment in the grant of equipment: https://gullfoss2.fcc.gov/prod/oet/cf/eas/reports/GenericSearch.cfm

  If this link does not work properly, please visit the FCC website (http://www.fcc.gov/) and follow the following steps to find the searching engine:

  FCC website ⟶ Office of Engineering Technology ⟶ Equipment Authorization Electronic Filing ⟶ Generic Search

Please notice that the output power listed in the grant uses different units depending on the type of the equipment, e.g.:

1. The output power for 802.11a/b/g/h equipment or similar equipment approved under §15.247 or §15.407 is listed as Conducted RF power. §15.247 or §15.407 limit the e.i.r.p. to 4 W, so this restriction is fulfilled.
2. The output power for Part 22 cellular equipment is listed as e.r.p. The relationship between e.r.p. and e.i.r.p. is the following one:

   e.i.r.p. = 1.64 x e.r.p.
3. The output power for Part 24 PCS equipment is listed as e.i.r.p.
4. For other type of equipment, please consult the distributor in order to assure the restriction is fulfilled.

Note: Definitions:
Effective Radiated Power (e.r.p.) (in a given direction): The product of the power supplied to the antenna and its gain relative to half-wave dipole in a given direction.
Equivalent Isotropically Radiated Power (e.i.r.p.) (in a given direction): The product of the power supplied to the antenna and its gain relative to an isotropic antenna.

The table below is excerpted from Table 1B of 47 CFR 1.1310 titled Limits for Maximum Permissible Exposure (MPE), Limits for General Population/Uncontrolled Exposure:

Frequency Range (MHz)	Power Density (mW/cm^2)
300 - 1500	f/1500
1500 - 100000	1.0

Table 10-1. Excerpt of Table 1B of 47 CFR 1.1310

The equipment WRAP Access Server equipment transmits in the 2400 - 2483.5 MHz frequency range, so the applicable MPE limit is 1 mW/cm^2. The equipment can be provided with up to 4 Bluetooth modules WT11# (FCC ID: QOQWT11):

Under the conditions stated above MPE limits can be guaranteed as the calculation below shows:

Example 10-1. 15.247 or 15.407 Compact Flash Card with maximum allowed e.i.r.p. of 4 W

Using Equation from page 18 of OET Bulletin 65, Edition 97-01:

$S_{\text{Compact Flash card}} = \text{Prad (e.i.r.p.)}_{\text{Compact Flash card}} / 4\pi R^2 = 4000\ mW/4\pi(20\ cm)^2$

$S_{\text{Compact Flash card}} = 0.795774\ \text{mW/cm}^2$

$S_{\text{Total}} = S_{\text{Bluetooth}} + S_{\text{Compact Flash card}} = 0.003481\ \text{mW/cm}^2 + 0.795774\ \text{mW/cm}^2$

$S_{\text{Total}} = 0.799255\ \text{mW/cm}^2 < 1\ \text{mW/cm}^2$

Example 10-2. Part 22 Compact Flash Card with maximum e.r.p. of 1.5 W (Category excluded of MPE evaluation according to §2.1091)

Using Equation from page 18 of OET Bulletin 65, Edition 97-01 and considering that e.i.r.p. = 1.64 x e.r.p.:

$S_{\text{Compact Flash card}} = \text{Prad (e.i.r.p.)}_{\text{Compact Flash card}} / 4\pi R^2 = 1500 \times 1.64\ \text{mW}/4\pi(20\ \text{cm})^2$

$S_{\text{Compact Flash card}} = 0.489401\ \text{mW/cm}^2$

$S_{\text{Total}} = S_{\text{Bluetooth}} + S_{\text{Compact Flash card}} = 0.003481\ \text{mW/cm}^2 + 0.489401\ \text{mW/cm}^2$

$S_{\text{Total}} = 0.492882\ \text{mW/cm}^2 < 1\ \text{mW/cm}^2$

Example 10-3. Part 24 Compact Flash Card with maximum e.r.p. of 3 W (Category excluded of MPE evaluation according to §2.1091)

Using Equation from page 18 of OET Bulletin 65, Edition 97-01 and considering that e.i.r.p. = 1.64 x e.r.p.:

$S_{\text{Compact Flash card}} = \text{Prad (e.i.r.p.)}_{\text{Compact Flash card}} / 4\pi R^2 = 3000 \times 1.64\ \text{mW} / 4\pi(20\text{cm})^2$

$S_{\text{Compact Flash card}} = 0.978803\ \text{mW/cm}^2$

$S_{\text{Total}} = S_{\text{Bluetooth}} + S_{\text{Compact Flash card}} = 0.003481\ \text{mW/cm}^2 + 0.978803\ \text{mW/cm}^2$

$S_{\text{Total}} = 0.982284\ \text{mW/cm}^2 < 1\ \text{mW/cm}^2$

WEEE Compliance

The crossed-out wheeled bin means that within the European Union the product must be taken to separate collection at the product end-of-life. Do not dispose of these products as unsorted municipal waste.

Appendix A. Directory Structure

```
Directory Tree                          Type Note
==============                          ==== ====
/                                       f    whole filesystem is root writable
|-- bin                                 f
|-- boot                                f
|-- dev                                 r
|   `-- shm                             r    ramdisk
|       |-- etc                         r    resolv.conf
|       |-- tmp                         r    /tmp
|       |   |-- obex                    r    obexserver dir
|       `-- var                         r    ramdisk part of /var
|           |-- lock                    r
|           |   `-- subsys              r
|           |-- log                     r
|           |-- run                     r
|           `-- empty                   r
|-- etc                                 f    system config and init scripts
|   |-- init.d -> rc.d/init.d           l
|   |-- ppp                             f
|   |   `-- peers                       f
|   |-- rc.d                            f
|   |   |-- init.d                      f
|   |   `-- rc3.d                       f
|   |-- rc3.d -> rc.d/rc3.d             l
|   |-- ssh                             f
|   `-- sysconfig                       f
|-- lib                                 f    system libraries
|   |-- iptables                        f
|   |-- pppd                            f
|   `-- modules                         f
|           `-- [module directories]    f
|-- mnt                                 f    mount points
|   |-- nfs                             f    empty mount point
|   `-- usb                             f    empty mount point
|-- proc                                p    proc filesystem
|-- root                                f    home directory of root
|-- sbin                                f
|-- sys                                 p    sys filesystem
|-- tmp -> dev/shm/tmp                  l    temporary data (ramdisk)
|-- usr                                 f
|   |-- bin                             f
|   |-- lib                             f
|   |   `-- gconv                       f
|   |-- libexec                         f
|   |-- local                           f    mount point for second flash
|   |-- sbin                            f
|   `-- share                           f
|       |-- tabset                      f
|       `-- terminfo                    f
|           |-- a                       f
```

```
|           |-- l                       f
|           |-- v                       f
|           `-- x                       f
`-- var                                 f
    |-- empty -> ../dev/shm/var/empty   f
    |-- lib                             f
    |   |-- b2b                         f
    |   |-- dpkg                        f
    |   |       `-- info                f
    |   `-- setup                       f
    |-- lock -> ../dev/shm/var/lock     l
    |-- log -> ../dev/shm/var/log       l    log files
    |-- run -> ../dev/shm/var/run       l
    |-- spool                           f
    |   `-- cron                        f
    |       `-- crontabs                f
    |-- tmp -> ../dev/shm/var/tmp       l
    `-- www                             f
        |-- cgi-bin                     f
        `-- html                        f    WWW pages

Types
=====

f = FLASH filesystem, read/write, files will be saved on power-down
r = RAM filesystem, read/write, files will be lost on power-down
l = symbolic link
p = proc/sys filesystem, can be used to configure Linux
```

Appendix B. Setup Options

B.1. Security settings

Submenu containing most important security settings, like passwords.

1. Root password [1rUj/KWS1$v3FZcBP.6HiN4f5PaATMq1]

   Password of "root" user, shown in encrypted form. The default is "buffy".

   To change the password, clear the field, enter a new password and click Save.
   Saving an empty field keeps the old password.

   Please note that the new password is shown in plain text only right after
   you have saved it. Later it is only shown encrypted, and there is no way
   to decrypt it. You must either remember it or change it again to
   something you do remember.

2. iWRAP password [buffy]

   The password required to be entered before any commands when communicating
   with iWRAP (the Bluetooth server). The default is "buffy".

   To change the password, clear the field, enter a new password and click Save.
   Saving an empty field keeps the old password.

   Please note that the new password is shown in plain text only right after
   you have saved it. Later it is only shown encrypted, and there is no way
   to decrypt it. You must either remember it or change it again to
   something you do remember.

   Use "-" to disable iWRAP password.

3. Do not require iWRAP password from local clients [Yes]

   Ask iWRAP password only from remote clients, not from local (127.0.0.1).

4. Bluetooth PIN code []

   This PIN code used when establishing connections. Up to 16 characters are
   significant.

   If there is no default PIN code set, Access Server does not require
   a PIN code when establishing connections.

   However, if there is no default PIN code set, but the other device
   requests a PIN code, "1234" is replied.

5. wpkgd autoinstall password []

   This is optional password to authenticate wpk autoinstall packets (wpk
   packets sent to the autoinstall directory, /tmp/obex by default).
   The password is shown encrypted here, if set. By default, it is not set.

   To change the password, clear the field, enter a new password and click Save.

   Please note that the new password is shown in plain text only right after

```
you have saved it. Later it is only shown encrypted, and there is no way
to decrypt it. You must either remember it or change it again to
something you do remember.

Use "-" do disable the password.

The password must match the authentication parameter in the "wpkg.pif"
file in the wpk packet. Otherwise the packet is not processed.

Syntax in the "wpkg.pif" file:
%wpkg-auth: auth
```

6. wpkgd hotplug password []

```
This is optional password to authenticate wpk installation packets
automatically run from USB memory dongles or Compact Flash memory cards.
The password is shown encrypted here, if set. By default, it is not set.

To change the password, clear the field, enter a new password and click Save.

Please note that the new password is shown in plain text only right after
you have saved it. Later it is only shown encrypted, and there is no way
to decrypt it. You must either remember it or change it again to
something you do remember.

Use "-" to disable the password.

The password must match the authentication parameter in the "wpkg.pif"
file in the wpk packet. Otherwise the packet is not processed.

Syntax in the "wpkg.pif" file:
%wpkg-auth: auth
```

7. Root user password for FTP [buffy]

```
Password of the "root" user for FTP connections.
```

8. Allow anonymous FTP login [Yes]

```
Whether "anonymous" FTP login is allowed or not.
```

9. WWW passwords [/etc/httpd.conf]

```
Access to WWW pages served by Access Server can be restricted using the
configuration file "httpd.conf", editable from here.

The file consists of lines in format "/dir:username:password". This
specifies that to view the WWW page at address "http://as-ip/dir", you
must enter username "username" and password "password".

More than one username can be defined for the same "/dir"
by adding multiple lines.

By default, this file specifies that only user "root" with password
"buffy" is allowed to access the WWW Setup.
```

B.2. Generic settings

Submenu containing generic settings.

1. Root password [1rUj/KWS1$v3FZcBP.6HiN4f5PaATMq1]

   Password of "root" user, shown in encrypted form. The default is "buffy".

   To change the password, clear the field, enter a new password and click Save. Saving an empty field keeps the old password.

   Please note that the new password is shown in plain text only right after you have saved it. Later it is only shown encrypted, and there is no way to decrypt it. You must either remember it or change it again to something you do remember.

2. Use local syslog service [Yes]

   This option determines whether the System Logger (syslogd) logs locally to /var/log/messages or not.

   Set this to No if you want to log to a remote syslog server.

3. IP address of the remote syslog server [192.168.42.1]

   The IP address of the device in the network to which the System Logger should log to.

   The remote device must be configured to accept syslogd connections from this Access Server. See the system logger documentation on the remote device for more information on how to configure that.

B.3. Network settings

Submenu containing network settings.

1. Hostname of the unit [wrap]

   The hostname of Access Server. Local applications will see this name. This name may be changed by dynamic network configuration.

2. Domain of the unit [localdomain]

   The domain name of Access Server. Local applications will see this name. This name may be changed by dynamic network configuration.

3. Enable Ethernet cable interface [Yes]

   Set this option to Yes if you want to have the Ethernet cable interface enabled.

   If you don’t use this interface, you may disable it to slightly increase security and system boot speed.

4. Enable Wi-Fi interface [Yes]

   Set this option to Yes if you want to have the Wi-Fi interface enabled (you can use the Wi-Fi interface with a supported Compact Flash Wi-Fi card or USB Wi-Fi dongle).

   If you don’t use this interface, you may disable it to slightly increase

security and system boot speed.

5. Enable GPRS interface [No]

   Set this option to Yes if you want to have the GPRS interface enabled.
   To use the interface, a supported Compact Flash GPRS card or a serial GPRS
   modem must be attached to Access Server.

6. Time server (rdate) []

   Hostname or IP address of the time server to be connected at system boot to
   retrieve correct time using the Time Protocol (RFC 868).

   NTP client is running by default, so rdate should not be needed at all.

7. Zeroconf interface [nap]

   Defines the interface in which Zeroconf is running.
   Possible interface names are "nap", "gn" and "none".

B.3.1. Default interface settings

Default interface settings. By default, Ethernet and Bluetooth
PAN-NAP interfaces are assigned to this interface.

1. Use dynamic network configuration [Yes]

   This option determines whether or not automatic configuration of the default
   network interface (nap) using DHCP should be attempted at boot. If set to
   no, you have to manually enter IP address and other network settings.

2. IP address [192.168.42.3]

   The IP address of Access Server.

3. Subnet mask [255.255.255.0]

   The network mask of Access Server.

4. IP address of the default gateway [192.168.42.254]

   The IP address of the default gateway in the LAN to which Access Server
   is connected.

5. List of name server IPs [192.168.42.1 192.168.42.2]

   The IP address(es) of the name servers, separated by space.

B.3.2. Ethernet cable settings

Ethernet cable settings.

1. Assign to default interface [Yes]

   Assigns Ethernet (eth0) to default interface (nap) with settings
   specified in Default interface settings.

   Do NOT set this to No if you don't know what you are doing. There
   is a high risk that you end up with invalid network settings if you
   do so.

   If you need to set a static IP address to Access Server, do it
   in the Default interface settings.

2. Use dynamic network configuration [Yes]

   ```
   Use dynamic network configuration (DHCP) on Ethernet interface when
   it is not assigned to the default interface.
   ```

3. IP address [192.168.43.3]

   ```
   IP address of the Ethernet interface when it is not assigned to the
   default interface and dynamic network configuration is not in use.
   ```

4. Subnet mask [255.255.255.0]

   ```
   Network mask of the Ethernet interface when it is not assigned to the
   default interface and dynamic network configuration is not in use.
   ```

B.3.3. Wi-Fi settings

```
Wi-Fi settings.
```

1. Act as a Wi-Fi Access Point [No]

   ```
   This option defines whether Access Server acts as a Wi-Fi Access
   Point when Wi-Fi is enabled.
   ```

2. ESSID []

   ```
   Access point network name (Service Set ID).
   ```

3. Nickname []

   ```
   The nickname, or station name.
   ```

4. WEP encryption key []

   ```
   WEP encryption key for Wi-Fi.

   Examples:
   10 hex digits:       "abcdef1234"
   26 hex digits:       "1234567890abcdef1234567890"
   or                   "1234-5678-90ab-cdef-1234-5678-90"
   5 ASCII characters:  "s:abcde"
   13 ASCII characters: "s:abcdefghijklm"
   ```

5. Extra commands for Access Point mode [/etc/sysconfig/ifup-wlan0]

   ```
   Extra commands for Access Point mode.
   ```

6. Assign to default interface [No]

   ```
   Assigns Wi-Fi to default interface with settings specified in
   Default interface settings.
   ```

7. Use dynamic network configuration [Yes]

   ```
   Use dynamic network configuration (DHCP) for Wi-Fi interface.
   ```

8. IP address [192.168.44.3]

   ```
   IP address of Wi-Fi interface.
   ```

9. Subnet mask [255.255.255.0]

   ```
   Subnet mask of Wi-Fi interface.
   ```

B.3.4. GPRS settings

```
GPRS settings.
```

1. Dial on demand [Yes]

   ```
   If this option is set to Yes, the GPRS link is not opened at boot time but
   when there is data to be transferred.
   ```

2. SIM card PIN code []

   ```
   PIN code of the SIM card in the GPRS modem.
   ```

3. Username [blue]

   ```
   Username for GPRS network. Contact your GSM operator for correct value.

   Some examples:

   Elisa/Finland:   blue
   Sonera/Finland:  blue
   Wataniya/Kuwait: blue
   Etisalat/UAE:    Mnet

   See also: http://www.kh-gps.de/gprsset.htm
   ```

4. Password [giga]

   ```
   Password for GPRS network. Contact your GSM operator for correct value.

   Some examples:

   Elisa/Finland:   giga
   Sonera/Finland:  giga
   Wataniya/Kuwait: giga
   Etisalat/UAE:    Mnet

   See also: http://www.kh-gps.de/gprsset.htm
   ```

5. Internet APN [internet]

   ```
   Internet APN for GPRS network. Contact your GSM operator for correct value.

   Some examples:

   Elisa/Finland:   internet
   Sonera/Finland:  internet
   Wataniya/Kuwait: action.wataniya.com
   Etisalat/UAE:    mnet

   See also: http://www.kh-gps.de/gprsset.htm
   ```

6. Extra parameters for pppd []

   ```
   Optional extra parameters for pppd. Use only if you know what you are doing.
   ```

B.4. Applications

Submenu containing settings of various applications.

1. Default startup applications []

   Change which applications are to be started at startup and which don't.

B.4.1. wpkgd settings

Submenu containing settings for wpkgd application.

1. wpkgd's autoinstall directory [/tmp/obex]

   wpkgd will automatically check this directory for wpk files containing
   software update packets.

   Use "/tmp/obex" if you want to allow updates via Bluetooth Object Push.
   Use empty to disable autoinstall.

2. Password for autoinstall packages []

   This is optional password to authenticate wpk autoinstall packets (wpk
   packets sent to the autoinstall directory, /tmp/obex by default).
   The password is shown encrypted here, if set. By default, it is not set.

   To change the password, clear the field, enter a new password and click Save.

   Please note that the new password is shown in plain text only right after
   you have saved it. Later it is only shown encrypted, and there is no way
   to decrypt it. You must either remember it or change it again to
   something you do remember.

   Use "-" do disable the password.

   The password must match the authentication parameter in the "wpkg.pif"
   file in the wpk packet. Otherwise the packet is not processed.

   Syntax in the "wpkg.pif" file:
   %wpkg-auth: auth

3. Delete processed autoinstall packages [Yes]

   If this option is set Yes, the wpk autoinstall packets are deleted
   after they have been processed.

4. Process hotplug packages [Yes]

   If this option is set to Yes, wpk packets are automatically processed
   from USB memory sticks or Compact Flash memory cards when they are
   plugged into Access Server.

5. Password for hotplug packages []

   This is optional password to authenticate wpk installation packets
   automatically run from USB memory dongles or Compact Flash memory cards.
   The password is shown encrypted here, if set. By default, it is not set.

   To change the password, clear the field, enter a new password and click Save.

```
Please note that the new password is shown in plain text only right after
you have saved it. Later it is only shown encrypted, and there is no way
to decrypt it. You must either remember it or change it again to
something you do remember.

Use "-" to disable the password.

The password must match the authentication parameter in the "wpkg.pif"
file in the wpk packet. Otherwise the packet is not processed.

Syntax in the "wpkg.pif" file:
%wpkg-auth: auth
```

6. Delete processed hotplug packages [No]

```
If this option is set Yes, the wpk packets are deleted
after they have been processed.
```

7. Extra parameters for wpkgd []

```
Optional extra command line parameters for wpkgd.

Please see wpkgd --help for detailed information on the options.
```

B.4.2. FTP server settings

```
Submenu containing settings for FTP server application.
```

1. Root user password [buffy]

```
Password of the "root" user for FTP connections.
```

2. Root user directory [/]

```
Root directory of the "root" user for FTP connections.
```

3. Root user instances [5]

```
Maximum number of simultaneous logins of the "root" user for FTP
connections.
```

4. Allow anonymous login [Yes]

```
Whether "anonymous" FTP login is allowed or not.
```

5. Anonymous user password [*]

```
Password of the "anonymous" user for FTP connections.

Use "*" to allow everything (aka anonymous login).
```

6. Anonymous user directory [/tmp/obex]

```
Root directory of the "anonymous" user for FTP connections.
```

7. Anonymous user instances [5]

```
Maximum number of simultaneous logins of the "anonymous" user for FTP
connections.
```

8. Allow anonymous user to do everything [No]

```
Whether "anonymous" user is allowed to do everything (all below) or not.
```

9. Allow anonymous user to download [Yes]

   Whether "anonymous" user is allowed to download files or not.

10. Allow anonymous user to upload [No]

    Whether "anonymous" user is allowed to upload files and make directories or not.

11. Allow anonymous user to overwrite [No]

    Whether "anonymous" user is allowed to overwrite existing files or not.

12. Allow anonymous user to multiple login [No]

    Whether "anonymous" user is allowed to multiple logins or not.

13. Allow anonymous user to erase [No]

    Whether "anonymous" user is allowed to erase files and directories or not.

14. Edit configuration file [/etc/ftpd.conf]

    Edit the self documented configuration file of the FTP server. Here you can change more advanced settings.

B.4.3. ObexSender settings

Submenu containing settings for ObexSender application.

1. Bluetooth friendly name [W$S_$p]

   The name shown when this device is found when inquired about by other Bluetooth devices. Following meta tags are available:

   $S : Hardware serial number, all ten digits
   $s : Hardware serial number, last three digits
   $P : Server port
   $p : Server port, last digit
   $H : Fully Qualified Domain Name (FQDN)
   $h : hostname
   $$: $

   For example, "Server_$p" would set the Bluetooth friendly name as "Server_1" for 1st baseband, "Server_2" for 2nd baseband and "Server_3" for 3rd baseband.

2. Delay between inquiries [10]

   Delay between inquiries (Bluetooth device discoveries) in seconds.

3. Delay between reply scans [10]

   Determines how often (in seconds) OBEX incoming directory (/tmp/obex) is scanned for remote requests. A low value increases CPU usage.

4. If previous was ok, timeout before sending again [36000]

   If a file has been successfully sent to a device, this timeout (in seconds) defines when content can be sent again to the same device.

5. If previous was reject, timeout before trying again [86400]

   If a file transmission to a device has failed or user has declined

```
the file, this timeout (in seconds) defines when ObexSender can
send content to the same device again.
```

6. Delay between retrying call [120]

```
When user doesn't accept or reject the file, ObexSender will try to
send the file again. This setting determines the timeout (in seconds)
before resend occurs. Default value is 120 seconds.

If you wish to disable this feature you can use the same value as in
"ok delay" or "reject delay", i.e. the two previous settings.
```

7. Delay after scanning [5]

```
When a remote request from user has been received, this setting
determines how long (in seconds) ObexSender will wait until the
response file is sent back to the user.

Default value is 5 seconds, because some mobile phones are not
able to receive files over Bluetooth until at least 5 seconds
has passed from sending.
```

8. Delay between multiple files [40]

```
If ObexSender has been configured to send multiple files, this
configuration sets the delay (in seconds) between the file
transmissions.
```

9. Minimum RSSI value before sending [-65]

```
The working range of ObexSender can be configured or limited with
this setting. When ObexSender searches for devices, the RSSI
(Receiver Signal Strength Indicator) value is also measured.
This value ranges from -128 to -1.

-128 means the signal strength is very weak. A connection attempt would
very likely fail.

-65 means the signal strength is ok. Connection can be created.
With Class 2 devices, like most mobile phones, this means the
phone is 10-20 meters away. A Class 1 device can be even more
than 100 meters away.

-30 to -1 means the signal is very strong. The devices are most likely
very close to each other (less than a meter away).
```

10. Logfile name [-]

```
Defines the path and name of the ObexSender log file
(for example "/usr/local/obexsender/obexsender.log").
Log file contains information about successful and unsuccessful
transmissions, timestamps and information about sent files.

You can also use an IP address of a log server, which must be another
Access Server running ObexSender.

Type "-" to use syslog.
```

11. Log prefix [-]

```
This prefix is put in front of every event in the log file.
Type "-" for none (default).
```

12. If sending was failure, log it too [Yes]

```
If this is enabled failed transmissions will be logged too.
```

13. Register to watchdog daemon [Yes]

```
If this is enabled, ObexSender will reboot Access Server
automatically if Bluetooth basebands have stopped responding.
```

14. iWRAP password [-]

```
iWRAP password. "-" for none (default).
```

15. Edit configuration file [/etc/obexsender.conf]

```
This link opens ObexSender configuration file
(/etc/obexsender.conf) and allows you to edit it manually.

It also allows you to change the settings that are not
configurable with Setup application.
```

16. Upload a new file [/usr/local/obexsender]

```
This link allows you to upload files into the ObexSender file directory.
```

17. List files [/usr/local/obexsender]

```
This link allows you to browse files on the ObexSender file system.
```

18. View log [-]

```
This link allows you to view ObexSender log file if it exists.
By default a summary of the logged events is displayed.
Detailed information is available by clicking the date links.
```

B.4.3.1. Delete log (confirm)

```
This link will delete the current log file after confirmation.
```

1. Delete log now! [/bin/true]

```
Delete ObexSender log file immediately!

WARNING: There is no confirmation for this!
```

B.4.4. SMS gateway settings

```
Submenu containing settings for SMS gateway application.
```

1. Modem device [/dev/ttyS0]

```
Modem device for SMS gateway.

/dev/ttyAT1 for user uart
/dev/ttyS0 for CF slot
```

2. Log file name [-]

```
The file to which the SMS gateway (smsgw) logs all traffic. Use /dev/null
for none, - for syslog, /var/log/smsgw.log if you want to save this
information. Be careful, however, not to fill the RAM file system (use a
```

```
cron job to free disk space from time to time).
```

3. SMSC number [+358405202000]

```
SMSC number. Contact your local GSM operator if you don't know the correct
value.

+358405202000 for Sonera/Finland
+358508771010 for Elisa/Finland
```

4. Edit configuration file [/etc/smsgw.conf]

```
Edit the self documented configuration file of the SMS gateway.
```

B.5. Bluetooth settings

```
Submenu containing all Bluetooth related settings.
```

1. iWRAP password [buffy]

```
The password required to be entered before any commands when communicating
with iWRAP (the Bluetooth server). The default is "buffy".

To change the password, clear the field, enter a new password and click Save.
Saving an empty field keeps the old password.

Please note that the new password is shown in plain text only right after
you have saved it. Later it is only shown encrypted, and there is no way
to decrypt it. You must either remember it or change it again to
something you do remember.

Use "-" to disable iWRAP password.
```

2. Do not require iWRAP password from local clients [Yes]

```
Ask iWRAP password only from remote clients, not from local (127.0.0.1).
```

3. Friendly name [W$S_$p]

```
The name shown when this device is found when inquired about by other
Bluetooth devices. Following meta tags are available:

$S : Hardware serial number, all ten digits
$s : Hardware serial number, last three digits
$P : Server port
$p : Server port, last digit
$H : Fully Qualified Domain Name (FQDN)
$h : hostname
$$ : $

For example, "Server_$p" would set the Bluetooth friendly name as
"Server_1" for 1st baseband, "Server_2" for 2nd baseband and
"Server_3" for 3rd baseband.
```

4. Connectable and discoverable mode [3]

```
This setting specifies whether this device is connectable and/or
discoverable or not by other Bluetooth devices.
```

When a device is connectable, other Bluetooth devices can make a Bluetooth connection to it. Before making a connection, the calling device must know the Bluetooth address of the device it is connecting to. The Bluetooth addresses can be found by making an inquiry. When a device is discoverable, it shows up in inquiries. Possible values for all combinations of these settings are:

0 : Not connectable, not discoverable
1 : Not connectable, discoverable
2 : Connectable, not discoverable
3 : Connectable and discoverable (default)

5. Master/slave role switch policy [1]

This setting specifies how local Bluetooth device should decide it's role. When a Bluetooth device calls another Bluetooth device, it is master by default and the answering device is slave. When the connection is being built, a role switch can be made. Normally, access point devices need to be the master, and therefore they require a master-slave switch when a new device is connecting. This is also how Access Server is configured by default. Otherwise Access server couldn't host the maximum number of slaves (7). Other possible combinations are:

0 : Allow switch when calling, don't request it when answering
1 : Allow switch when calling, request it when answering (default)
2 : Don't allow switch when calling, request it when answering

If you have problems with connecting to Access Server, it might be because your client device does not support the master/slave switch. In this case, set this setting to 0.

6. Default PIN code []

This PIN code used when establishing connections. Up to 16 characters are significant.

If there is no default PIN code set, Access Server does not require a PIN code when establishing connections.

However, if there is no default PIN code set, but the other device requests a PIN code, "1234" is replied.

7. Power save mode and parameters [4]

The power save mode used by default for all connections. Possible settings are:

0 : Active.
1 : Park: Round-robin.
2 : Park: Idle.
3 : Sniff: All
4 : Sniff: Idle (default).

"Active" means that no power saving is in use.
"Sniff: All" means that the connections are kept in sniff mode always.

"Sniff: Idle" means that a connection is switched to sniff mode after it has not transmitted data for some time (2 seconds by default). When data transmission resumes, switch to active mode is made.

Park modes are generally not useful. See User's and Developer's Guide and Bluetooth specification for more information.

8. Use literal replies in SDP [Yes]

   If enabled, some SDP result codes will have literal values instead of numeric values.

9. Optional command line parameters []

   Optional extra command line startup parameters for the iWRAP servers.

10. Edit startup script [/etc/bluetooth.conf]

    Opens iWRAP configuration file (/etc/bluetooth.conf) for editing.

    You can append extra iWRAP commands to that file. iWRAP servers process the file each time they start. See the User's and Developer's Guide for iWRAP command reference.

B.5.1. Bluetooth profiles

Submenu for the settings of all supported Bluetooth profiles.

1. Enable lan access profile [No]

   Whether or not the LAN Access Profile is enabled.

2. Enable PAN user profile [No]

   Whether or not the PAN User Profile is enabled.

3. Enable PAN generic networking profile [No]

   Whether or not the PAN Generic Networking Profile is enabled.

4. Enable PAN network access point profile [No]

   Whether or not the PAN Network Access Point Profile is enabled.

5. Enable object push profile [Yes]

   Whether or not the Object Push Profile is enabled.

6. Enable file transfer profile [Yes]

   Whether or not the File Transfer Profile is enabled.

B.5.1.1. Lan access profile settings

Submenu containing LAN Access Profile settings.

1. Login name and password []

   The login name and password required from LAN access clients. Must be entered as a single string, separated with a space. For example: guest buffy

   If empty (default), no login is required.

2. Service name (shown in SDP) [Lan Access]

```
The name of the LAN Access Profile service shown in the Service Discovery.
```

3. Defaultroute modification policy [0]

```
How the LAN Access Profile should modify the defaultroute in routing tables:

0: Do not alter defaultroute (default)
1: When acting as a LAP client, set defaultroute according to the LAP server
2: When acting as a LAP server, set defaultroute according to the LAP client
3: Set defaultroute according to the LAP server/client connected
```

4. First IP for LAP clients [192.168.160.0]

```
This defines the C-class of IP addresses to be used in point-to-point
connections between Access Server and LAP clients.

Full C-class is required: use "x.y.z.0".
```

B.5.1.2. PAN user profile settings

```
Submenu containing Personal Area Network User Profile settings.
```

1. Service name (shown in SDP) [PAN User]

```
The name of the PAN User Profile service shown in the Service Discovery.
```

2. Enable zeroconf when calling [No]

```
Enable ZeroConf protocol for outgoing PANU connections.
```

3. Enable zeroconf when answering [No]

```
Enable ZeroConf protocol for incoming PANU connections.
```

B.5.1.3. PAN generic networking profile settings

```
Submenu containing Personal Area Network Generic Networking Profile
settings.
```

1. Service name (shown in SDP) [Generic Networking]

```
The name of the PAN Generic Networking Profile service shown in
the Service Discovery.
```

2. Use dynamic network configuration for local IP address [No]

```
Whether or not DHCP is used for configuring local IP Address. Enable only if
you are connecting this PAN-GN to another PAN-GN that will provide the IP
configuration.
```

3. Local GN interface IP address [192.168.161.1]

```
The IP address for the local GN interface.
```

4. Local GN interface subnet mask [255.255.255.0]

```
The netmask for the local GN interface.
```

5. Start DHCP server for remote users [Yes]

```
Whether or not this device should start DHCP for remote devices connecting
to this PAN-GN. Disabled if "Use dynamic network configuration for local IP
address" is used.
```

6. First IP for lease block [192.168.161.2]

First IP address of the lease block.

7. Last IP for lease block [192.168.161.254]

   Last IP address of the lease block.

8. Subnet of lease block [255.255.255.0]

   Subnet mask of the lease block.

9. Lease time [86400]

   Lease time in seconds.

B.5.1.4. PAN network access point profile settings

Submenu containing Personal Area Network Network Access Point Profile settings.

1. Service name (shown in SDP) [Network Access]

   The name of the Bluetooth PAN Network Access Point Profile service shown in the Service Discovery.

B.5.1.5. Serial port profile settings

Submenu containing the Bluetooth Serial Port Profile settings.

The profile itself is enabled and disabled by switching "serialport" application "on" or "off" from the menu:
Setup -> Applications -> Default bootup applications.

1. Act as the calling device [No]

   Whether this device should act as the calling device (DevA) or the answering device (DevB).

2. BPS rate [115200]

   The bits-per-second rate of the connection. Possible values are:
   300, 1200, 2400, 4800, 9600, 19200, 38400, 57600, 115200, 230400, and 460800.

3. Data bits [8]

   The number of data bits in the connection. Possible values are:
   5, 6, 7, and 8.

4. Parity [0]

   The parity bit setting of the connection. Possible values are:

   0: No Parity (default)
   1: Odd Parity
   2: Even Parity

5. Stop bits [1]

   The number of stop bits in the connection. Possible values are 1 and 2.

6. Hardware flow control (RTS/CTS) [Yes]

   Whether or not the hardware flow control is used.

7. Software flow control (XON/XOFF) [No]

   Whether or not the software flow control is used.

8. Bluetooth address of the remote device [00:07:80:80:bf:01]

   ```
   The Bluetooth address of the device to be contacted. If the local device
   is configured as DevA, this is the DevB it tries to connect.
   ```

9. Service channel [2]

   ```
   In DevA (call) mode:   The Bluetooth RFCOMM channel of the remote device.

   In DevB (answer) mode: The Bluetooth RFCOMM channel of the local device.
   ```

10. Service name (shown in SDP) [Serial Port]

    ```
    The name of the Bluetooth Serial Port Profile service shown in the Service
    Discovery.
    ```

11. Optional command line parameters []

    ```
    Optional extra parameters for the Access Server Serial Port profile
    application. Currently the supported parameters are:

    --device dev     Device, if not the user port (/dev/ttyS0 for CF Card)
    --msc            Enables transmitting of DCD/DSR Modem Status Control signals.
    --nobuffer       Discard data if no Bluetooth connection, do not buffer it.
    ```

B.5.1.6. Object push profile settings

```
This submenu contains Bluetooth Object Push Profile settings.
```

1. Service name (shown in SDP) [Object Push]

   ```
   The name of the Object Push Profile service shown in the Service Discovery.
   ```

B.5.1.7. File tranfer profile settings

```
This submenu contains Bluetooth File Transfer Profile settings.
```

1. Service name (shown in SDP) [File Transfer]

   ```
   The name of the File Transfer Profile shown in the Service Discovery.
   ```

B.6. Advanced settings

```
Submenu containing advanced settings of Access Server.
```

1. System startup script [/etc/rc.d/rc.local]

   ```
   This is the last initialization script executed at system startup.

   By default, the script /etc/rc.d/rc.local just turns off all LEDs to
   indicate the startup has finished. If you want to initialize something
   automatically at every boot, or start up your own applications,
   you should add the required commands to this file.

   Remember to start your programs to the background. Example:
   /usr/local/bin/myapp &

   If you do not start the programs to the backgroud, you will not able
   to access the management console using a serial cable.
   ```

2. Default user profile [/etc/profile]

```
Edit the file containing the default user profile settings.
```

3. WWW passwords [/etc/httpd.conf]

```
Access to WWW pages served by Access Server can be restricted using the
configuration file "httpd.conf", editable from here.

The file consists of lines in format "/dir:username:password". This
specifies that to view the WWW page at address "http://as-ip/dir", you
must enter username "username" and password "password".

More than one username can be defined for the same "/dir"
by adding multiple lines.

By default, this file specifies that only user "root" with password
"buffy" is allowed to access the WWW Setup.
```

4. Setup access [/etc/setup.conf]

```
The "/etc/setup.conf" file can be used to give different access rights to
different users of the WWW Setup.

The file consist of lines in following format:
example.tag +user1 +user2 -user3 -user4

This will allow (+) access to tag "example.tag" for "user1" and "user2"
and denies (-) access from "user3" and "user4". You can find the tags
from the output of
Setup -> Advanced -> System Information -> Collect info for support request

For example, the tag of this setting is advanced.setupconf. If you
have created another user "guest" in /etc/httpd.conf that can access
"/setup", you can deny that user from changing the Setup access settings
with following line in this file:

advanced.setupconf -guest
```

5. Edit other configuration files []

```
From this menu you can edit any files located in Access Server file system.
You can for example create "/var/spool/cron/crontabs/root" file for
configuring the cron daemon.
```

6. Browse files []

```
Browse files stored in Access Server.
```

7. Find other Access Servers [/usr/sbin/finder]

```
Find other Access Servers.
```

8. Inquiry for Bluetooth devices [/usr/bin/btcli inquiry]

```
Inquiry for other Bluetooth devices.
```

9. Upload a software update [/tmp/obex]

```
Upload a software update file (*.wpk).

Access Server supports a special management packet format (wpk), which
```

can be used to update Access Server software components or to install custom software and configuration files. Please consult User's and Developer's Guide for more information.

B.6.1. System information

This submenu contains tools to retrieve system status information.

1. Hardware information

   Displays hardware and software identification information (output of command "wrapid").

2. List installed software components [/usr/bin/dpkg -l]

   Lists currenty installed software components and their version numbers.

3. List running processes [/bin/ps ww]

   Lists running processes.

4. List memory status [/usr/bin/free]

   Lists memory status.

5. List free disk space [/bin/df -h]

   Lists free disk space.

6. Show system log file [/var/log/messages]

   Shows system log file.

7. Show system boot log file [/var/log/dmesg]

   Shows system boot log.

8. Collect info for support request [/usr/sbin/supportinfo]

   This page contains collectively all the system status and configuration information.

   Include this information when sending a support request to support@bluegiga.com

   WARNING: All classified information, like passwords, should be automatically excluded. It is still recommended to manually check that all such information is really removed.

B.6.2. Reboot system (confirm)

Reboot Access Server. Confirmation will be asked.

1. Reboot now! [/sbin/reboot]

   Reboot Access Server immediately!

   WARNING: There is no confirmation for this!

B.7. Summary of Setup Options

```
Security settings
    Root password                          [$1$rUj/KWS1$v3FZcBP.6HiN4f5PaATMq1]
```

```
    iWRAP password                          [buffy]
    Do not require iWRAP password from local clients          [Yes]
    Bluetooth PIN code                      []
    wpkgd autoinstall password              []
    wpkgd hotplug password                  []
    Root user password for FTP              [buffy]
    Allow anonymous FTP login               [Yes]
    WWW passwords                           [/etc/httpd.conf]

Generic settings
    Root password                           [$1$rUj/KWS1$v3FZcBP.6HiN4f5PaATMq1]
    Use local syslog service                [Yes]
    IP address of the remote syslog server          [192.168.42.1]

Network settings
    Hostname of the unit                    [wrap]
    Domain of the unit                      [localdomain]
    Default interface settings
        Use dynamic network configuration           [Yes]
        IP address                          [192.168.42.3]
        Subnet mask                         [255.255.255.0]
        IP address of the default gateway           [192.168.42.254]
        List of name server IPs             [192.168.42.1 192.168.42.2]
    Enable Ethernet cable interface         [Yes]
    Ethernet cable settings
        Assign to default interface         [Yes]
        Use dynamic network configuration           [Yes]
        IP address                          [192.168.43.3]
        Subnet mask                         [255.255.255.0]
    Enable Wi-Fi interface                  [Yes]
    Wi-Fi settings
        Act as a Wi-Fi Access Point         [No]
        ESSID                               []
        Nickname                            []
        WEP encryption key                  []
        Extra commands for Access Point mode        [/etc/sysconfig/ifup-wlan0]
        Assign to default interface         [No]
        Use dynamic network configuration           [Yes]
        IP address                          [192.168.44.3]
        Subnet mask                         [255.255.255.0]
    Enable GPRS interface                   [No]
    GPRS settings
        Dial on demand                      [Yes]
        SIM card PIN code                   []
        Username                            [blue]
        Password                            [giga]
        Internet APN                        [internet]
        Extra parameters for pppd           []
    Time server (rdate)                     []
    Zeroconf interface                      [nap]

Applications
    Default startup applications            []
```

```
    wpkgd settings
        wpkgd's autoinstall directory     [/tmp/obex]
        Password for autoinstall packages           []
        Delete processed autoinstall packages       [Yes]
        Process hotplug packages          [Yes]
        Password for hotplug packages     []
        Delete processed hotplug packages           [No]
        Extra parameters for wpkgd        []
    FTP server settings
        Root user password                [buffy]
        Root user directory               [/]
        Root user instances               [5]
        Allow anonymous login             [Yes]
        Anonymous user password           [*]
        Anonymous user directory          [/tmp/obex]
        Anonymous user instances          [5]
        Allow anonymous user to do everything       [No]
        Allow anonymous user to download  [Yes]
        Allow anonymous user to upload    [No]
        Allow anonymous user to overwrite           [No]
        Allow anonymous user to multiple login      [No]
        Allow anonymous user to erase     [No]
        Edit configuration file           [/etc/ftpd.conf]
    ObexSender settings
        Bluetooth friendly name           [W$S_$p]
        Delay between inquiries           [10]
        Delay between reply scans         [10]
        If previous was ok, timeout before sending again      [36000]
        If previous was reject, timeout before trying again   [86400]
        Delay between retrying call       [120]
        Delay after scanning              [5]
        Delay between multiple files      [40]
        Minimum RSSI value before sending           [-65]
        Logfile name                      [-]
        Log prefix                        [-]
        If sending was failure, log it too          [Yes]
        Register to watchdog daemon       [Yes]
        iWRAP password                    [-]
        Edit configuration file           [/etc/obexsender.conf]
        Upload a new file                 [/usr/local/obexsender]
        List files                        [/usr/local/obexsender]
        View log                          [-]
        Delete log (confirm)
            Delete log now!               [/bin/true]
    SMS gateway settings
        Modem device                      [/dev/ttyS0]
        Log file name                     [-]
        SMSC number                       [+358405202000]
        Edit configuration file           [/etc/smsgw.conf]

Bluetooth settings
    iWRAP password                        [buffy]
    Do not require iWRAP password from local clients          [Yes]
```

```
Friendly name                          [W$S_$p]
Connectable and discoverable mode      [3]
Master/slave role switch policy        [1]
Default PIN code                       []
Power save mode and parameters         [4]
Use literal replies in SDP             [Yes]
Optional command line parameters       []
Edit startup script                    [/etc/bluetooth.conf]
Bluetooth profiles
    Enable lan access profile          [No]
    Lan access profile settings
        Login name and password        []
        Service name (shown in SDP)    [Lan Access]
        Defaultroute modification policy        [0]
        First IP for LAP clients       [192.168.160.0]
    Enable PAN user profile            [No]
    PAN user profile settings
        Service name (shown in SDP)    [PAN User]
        Enable zeroconf when calling   [No]
        Enable zeroconf when answering          [No]
    Enable PAN generic networking profile       [No]
    PAN generic networking profile settings
        Service name (shown in SDP)    [Generic Networking]
        Use dynamic network configuration for local IP address      [No]
        Local GN interface IP address           [192.168.161.1]
        Local GN interface subnet mask          [255.255.255.0]
        Start DHCP server for remote users      [Yes]
        First IP for lease block       [192.168.161.2]
        Last IP for lease block        [192.168.161.254]
        Subnet of lease block          [255.255.255.0]
        Lease time                     [86400]
    Enable PAN network access point profile     [No]
    PAN network access point profile settings
        Service name (shown in SDP)    [Network Access]
    Serial port profile settings
        Act as the calling device      [No]
        BPS rate                       [115200]
        Data bits                      [8]
        Parity                         [0]
        Stop bits                      [1]
        Hardware flow control (RTS/CTS)         [Yes]
        Software flow control (XON/XOFF)        [No]
        Bluetooth address of the remote device  [00:07:80:80:bf:01]
        Service channel                [2]
        Service name (shown in SDP)    [Serial Port]
        Optional command line parameters        []
    Enable object push profile         [Yes]
    Object push profile settings
        Service name (shown in SDP)    [Object Push]
    Enable file transfer profile       [Yes]
    File tranfer profile settings
        Service name (shown in SDP)    [File Transfer]
```

```
Advanced settings
    System startup script                [/etc/rc.d/rc.local]
    Default user profile                 [/etc/profile]
    WWW passwords                        [/etc/httpd.conf]
    Setup access                         [/etc/setup.conf]
    Edit other configuration files       []
    Browse files                         []
    Find other Access Servers            [/usr/sbin/finder]
    Inquiry for Bluetooth devices        [/usr/bin/btcli inquiry]
    Upload a software update             [/tmp/obex]
    System information
        Hardware information
        List installed software components          [/usr/bin/dpkg -l]
        List running processes           [/bin/ps ww]
        List memory status               [/usr/bin/free]
        List free disk space             [/bin/df -h]
        Show system log file             [/var/log/messages]
        Show system boot log file        [/var/log/dmesg]
        Collect info for support request [/usr/sbin/supportinfo]
    Reboot system (confirm)
        Reboot now!                      [/sbin/reboot]
```

Appendix C. Open Source Software Licenses

Some Access Server software components are licensed under the terms and conditions of one or more open source licenses, listed in Table C-1 below.

License Appreviation	Description	URL
CMU/UCD	Carnegie Mellon University & Regents of the University of California's BSD style license (in net-snmp)	
GPL1	GNU General Public License Version 1, February 1989	http://www.fsf.org/licenses/info/GPLv1.html
GPL2	GNU General Public License Version 2, June 1991	http://www.opensource.org/licenses/gpl-license.php
GPL2+	GNU General Public License Version 2 or later	http://www.opensource.org/licenses/gpl-license.php
LGPL2	GNU Library General Public License Version 2, June 1991	http://www.gnu.org/copyleft/lgpl.html
LGPL2.1	GNU Lesser General Public License Version 2.1, February 1999	http://www.opensource.org/licenses/lgpl-license.php
BSD	Revised BSD License (without the advertising clause)	http://www.opensource.org/licenses/bsd-license.php
BSDorig	Original BSD License (with the advertising clause)	http://www.fsf.org/licenses/info/BSD_4Clause.html
MIT	MIT License (only one version exist, also known as X11 style license)	
MPL1.1	Mozilla Public License Version 1.1	http://www.mozilla.org/MPL/
OpenSSL	OpenSSL License (similar to BSDorig)	http://www.openssl.org/source/license.html
SSLeay	SSLeay License (similar to BSDorig)	http://www.openssl.org/source/license.html
ZLIB	ZLIB License (only one version exist)	http://www.gzip.org/zlib/zlib_license.html

Table C-1. Open Source Licenses in Access Server Software Components

The details of the open source software components and the license under which they are distributed are listed below in Table C-2. Software components not listed are licensed under Bluegiga's License Agreement.

Software Component	Version	License	Source URL
Das U-Boot	1.0.0 and git-060720	GPL2	http://sourceforge.net/projects/u-boot/
The bootloader. Initialized system, holds system configuration, loads and launches the Linux kernel.			

Software Component	Version	License	Source URL
Kernel			
Linux kernel	2.6.17	GPL2	http://www.kernel.org/
The Access Server kernel, responsible for resource allocation, low-level hardware interfaces, security etc.			
kernel at91 patches	2.6.17	GPL2	http://maxim.org.za/AT91RM9200/2.6/
ARM-Linux patches for the Linux kernel.			
Userland			
bash	2.05b	GPL1 & GPL2	http://www.gnu.org/software/bash/bash.html
GNU Project's Bourne Again SHell, interactive shell with Bourne shell syntax.			
binutils	2.15	GPL2 & LGPL2	http://www.gnu.org/software/binutils/
GNU Binutils, collection of binary tools, like GNU linker and GNU assembler.			
bridge-utils	0.9.6	GPL2	http://bridge.sourceforge.net/
Linux Ethernet bridging utilities, needed to manage bridging for WRAP Bluetooth PAN profiles and WLAN Access Point functionality.			
busybox	1.2.1	GPL2+	http://www.busybox.net/
Provides tens of general userland utilities.			
bzip2	1.0.3	GPLorig	http://www.bzip.org/
Compression library.			
crosstool	0.42	GPL2	http://kegel.com/crosstool/
GCC build script.			
e3	2.6.2	GPL2	http://www.sax.de/~adlibit/
Small text editor with different keybindings.			
ed	0.2	GPL2	http://www.gnu.org/software/ed/ed.html
An 8-bit clean, POSIX-compliant line editor.			
gcc	3.4.5	GPL2 & LGPL2	http://gcc.gnu.org/
GNU C/C++ compiler and related tools.			
gdb	6.4	GPL2 & LGPL2	http://www.gnu.org/software/gdb/gdb.html
GNU debugger.			
glibc	2.3.6	GPL2 & LGPL2.1	http://www.gnu.org/software/libc/libc.html
GNU C Library.			
hostap-utils	0.3.7	GPL2	http://hostap.epitest.fi/
Utility programs for managing hostap-driver.			
iptables	1.3.4	GPL2	http://www.netfilter.org/

Software Component	Version	License	Source URL
Administration tool for the Linux kernel IP packet filter.			
make	3.81	GPL2	http://www.gnu.org/software/make/
The Make.			
maradns	1.2.0.07.6	BSD	http://www.maradns.org/
DNS server.			
libpcap	0.9.4	BSD	http://www.tcpdump.org/
Provides portable framework for low-level network monitoring. Needed by tcpdump.			
lrzsz	0.12.20	GPL2	http://www.ohse.de/uwe/software/lrzsz.html
Provides X/Y/Zmodem download/upload tools.			
ncurses	5.3	MIT	http://www.gnu.org/software/ncurses/ncurses.html
Library for displaying and updating text on text-only terminals.			
netkit-ftp	0.17	BSDorig	ftp://ftp.uk.linux.org/pub/linux/Networking/netkit/
FTP client application.			
net-snmp	5.2.rc4	CMU/USD & BSD	http://www.net-snmp.org/
Suite of applications used to implement SNMP v1, SNMP v2c and SNMP v3 using both IPv4 and IPv6.			
ntpclient	2003_194	GPL2	http://doolittle.faludi.com/ntpclient/
NTP (RFC-1305) client.			
openssl	0.9.8a	OpenSSL & SSLeay	http://www.openssl.org/
Toolkit implementing SSL v2/v3, TLS v1 and general purpose cryptography library.			
openssh	4.5p1	BSD	http://www.openssh.com/
OpenSSH suite; server and client utilities.			
openvpn	2.0.5	GPL2	http://openvpn.net/
An Open Source VPN daemon.			
pcmciautils	012	GPL2	http://kernel.org/pub/linux/utils/kernel/pcmcia/pcmcia.html
A suite of userspace tools for PCMCIA support in the Linux 2.6 kernel.			
perl	5.8.8	GPL2	http://www.perl.org/
A programming language.			
picocom	1.4	GPL2	http://efault.net/npat/hacks/picocom/
Minimal dumb-terminal emulation program.			
ppp	2.4.3	BSD & BSDorig & GPL2 & ZLIB	http://ppp.samba.org/

Software Component	Version	License	Source URL
Point-to-Point Protocol userland driver.			
ppp-dhcpc	for pppd 2.4.2	GPL2	ben at netservers.co.uk
DHCP plugin for PPP.			
readline	4.3	GPL2	http://cnswww.cns.cwru.edu/php/chet/readline/rltop.html
GNU Readline library, providing set of functions for use by applications that allow users to edit command lines as they are typed in.			
strace	4.5.14	GPL2	http://www.liacs.nl/~wichert/strace/
System call trace, i.e. a debugging tool.			
stupid-ftpd	1.4beta	GPL2	http://stupid-ftpd.sourceforge.net/
Simple FTP server.			
sysfsutils	2.0.0	GPL2	http://linux-diag.sourceforge.net/Sysfsutils.html
These are a set of utilites built upon sysfs, a new virtual filesystem in Linux kernel versions 2.5+ that exposes a system's device tree.			
termcap	2.0.8	GPL2	https://www.redhat.com/fedora/
Basic system library needed to access the termcap database.			
tftp-hpa	0.42	BSD	http://www.kernel.org/pub/software/network/tftp/
TFTP client and server.			
tcpdump	3.9.4	BSD	http://www.tcpdump.org/
Utility to monitor network traffic.			
wireless_tools	28	GPL2	http://www.hpl.hp.com/personal/Jean_Tourrilhes/Linux/Tools.html
Package containing utilities to manage Wireless LAN specific parameters.			
zlib	1.2.3	ZLIB	http://www.gzip.org/zlib/
General purpose compression library.			

Table C-2. Access Server Open Source Software Components and Their Licences

Appendix D. Supported Hardware

Connector	Type	Card	Note
CF	GPRS	Enfora GSM/GPRS Compact Flash Card (GSM 0110)	Multislot class 8.
CF	GPRS	Anycom GS-320 Tri-Band GPRS CF Card	Multislot class 10.
CF	GPRS	AudioVox RTM 8000	Multislot class 8, "same" HW as Fujitsu.
CF	GPRS	Fujitsu Siemens Connect2Air 3GSM	Multislot class 8, "same" HW as Audiovox.
CF	GPS	Pretec CompactGPS™	
CF	WiFi	Ambicom Wireless CompactFlash Card (WL1100C-CF)	Supports both client and access point modes
CF	WiFi	D-Link Air Wireless Network DCF-660W	Seen shipping with 1.7.4 firmware (can be access point without upgrade)
CF	WiFi	Linksys Instant Wireless WCF-12	
CF	WiFi	SMC Networks WLAN EZ Connect	Does not support firmware upgrade
CF	Memory	Any vendor	If you find a card that does not work, please contact <support@bluegiga.com>.
USB	EDGE/GPRS/GSM	Sagem Samba 75	Seen as modem device /dev/ttyACM0
USB	Memory	Any vendor	If you find a dongle that does not work, please contact <support@bluegiga.com>.

Table D-1. Supported Hardware by Access Server