CHAPTER 2

Preparing a System for Asterisk

Very early on, I knew that someday in some "perfect" future out there over the horizon, it would be commonplace for computers to handle all of the necessary processing functionality internally, making the necessary external hardware to connect up to telecom interfaces VERY inexpensive and in some cases trivial.

—Jim Dixon, "The History of Zapata Telephony and How It Relates to the Asterisk PBX"

By this point, you must be anxious to get your Asterisk system up and running. If you are building a hobby system, you can probably jump right to the next chapter and begin the installation. For a mission-critical deployment, however, some thought must be given to the environment in which the Asterisk system will run. Make no mistake: Asterisk, being a very flexible piece of software, will happily and successfully install on nearly any Linux platform you can conceive of, and several non-Linux platforms as well.[*] However, to arm you with an understanding of the type of operating environment Asterisk will really thrive in, this chapter will discuss issues you need to be aware of in order to deliver a reliable, well-designed system.

In terms of its resource requirements, Asterisk's needs are similar to those of an embedded, real-time application. This is due in large part to its need to have priority access to the processor and system buses. It is therefore imperative that any functions on the system not directly related to the call-processing tasks of Asterisk be run at a low priority, if at all. On smaller systems and hobby systems, this might not be as much of an issue. However, on high-capacity systems, performance shortcomings will manifest as audio quality problems for users, often experienced as echo, static,

[*] People have successfully compiled and run Asterisk on WRAP boards, Linksys WRT54G routers, Soekris systems, Pentium 100s, PDAs, Apple Macs, Sun SPARCs, laptops, and more. Of course, whether you would *want* to put such a system into production is another matter entirely. (Actually, the AstLinux distribution, by Kristian Kielhofner, runs very well indeed on the Soekris 4801 board. Once you've grasped the basics of Asterisk, this is something worth looking into further. Check out *http://www.astlinux.org*.)

and the like. The symptoms will resemble those experienced on a cell phone when going out of range, although the underlying causes will be different. As loads increase, the system will have increasing difficulty maintaining connections. For a PBX, such a situation is nothing short of disastrous, so careful attention to performance requirements is a critical consideration during the platform selection process.

Table 2-1 lists some very basic guidelines that you'll want to keep in mind when planning your system. The next section takes a close look at the various design and implementation issues that will affect its performance.

Table 2-1. System requirement guidelines

Purpose	Number of channels	Minimum recommended
Hobby system	No more than 5	400-MHz x86, 256 MB RAM
SOHO[a] system	5 to 10	1-GHz x86, 512 MB RAM
Small business system	Up to 15	3-GHz x86, 1 GB RAM
Medium to large system	More than 15	Dual CPUs, possibly also multiple servers in a distributed architecture

[a] Small Office/home Office—less than three lines and five sets.

With large Asterisk installations, it is common to deploy functionality across several servers. One or more central units will be dedicated to call processing; these will be complemented by one or more ancillary servers handling peripherals (such as a database, voicemail, conferencing, management, a web interface, a firewall, and so on). As is true in most Linux environments, Asterisk is well suited to growing with your needs: a small system that used to be able to handle all your call-processing and peripheral tasks can be distributed between several servers when increased demands exceed its abilities. Flexibility is a key reason why Asterisk is extremely cost-effective for rapidly growing businesses—there is no effective maximum or minimum size to consider when budgeting the initial purchase. While some scalability is possible with most telephone systems, we have yet to hear of one that can scale as inexpensively as Asterisk. Having said that, distributed Asterisk systems are not simple to design—this is not a task for someone new to Asterisk.*

Server Hardware Selection

The selection of a server is both simple and complicated: simple because, really, any x86-based platform will suffice; but complicated because the reliable performance of your system will depend on the care that is put into the platform design. When

* If you are sure that you need to set up a distributed Asterisk system, you will want to study the DUNDi protocol. You should probably get the interest of the *Asterisk-Users* mailing list as well, but be sure to wear your flame-retardant suit; for some reason, this subject can spur a heated (but generally very educational) debate.

selecting your hardware, you must carefully consider the overall design of your system and what functionality you need to support. This will help you determine your requirements for the CPU, motherboard, and power supply. If you are simply setting up your first Asterisk system for the purpose of learning, you can safely ignore the information in this section. If, however, you are building a mission-critical system suitable for deployment, these are issues that require some thought.

Performance Issues

Among other considerations, when selecting the hardware for an Asterisk installation you must bear in mind this critical question: how powerful must the system be? This is not an easy question to answer, because the manner in which the system is to be used will play a big role in the resources it will consume. There is no such thing as an Asterisk performance-engineering matrix, so you will need to understand how Asterisk uses the system in order to make intelligent decisions about what kinds of resources will be required. You will need to consider several factors, including:

The maximum number of concurrent connections the system will be expected to support
: Each connection will increase the workload on the system.

The percentage of traffic that will require processor-intensive Digital Signal Processing (DSP) of compressed codecs (such as G.729 and GSM)[*]
: The DSP work that Asterisk performs in software can have a staggering impact on the number of concurrent calls it will support. A system that can happily handle 50 concurrent G.711 calls can be brought to its knees by a request to conference together 10 G.729 compressed channels.

Whether conferencing will be provided, and what level of conferencing activity is expected
: Will the system be used heavily? Conferencing requires the system to transcode and mix each individual incoming audio stream into multiple outgoing streams. Mixing multiple audio streams in near-real-time can place an enormous load on the CPU.

Echo cancellation[†]
: Echo cancellation may be required on any calls where a Public Switched Telephone Network (PSTN) interface is involved. Since echo cancellation is a mathematical function—the more of it the system has to perform, the higher the load on the CPU will be.

Dialplan scripting logic
: Whenever Asterisk has to pass call control to an external program, there is a performance penalty. As much logic as possible should be built into the dialplan. If external scripts are used, they should be designed with performance and efficiency as important goals.

[*] We'll talk more about G.729, GSM, G.711, and many other codecs in Chapter 8.

[†] Do not fear. Echo cancellation is another topic for Chapter 8.

As for the exact performance impact of these factors, the jury's still out. The effect of each is known in general terms, but an accurate performance calculator has not yet been successfully defined. This is partly because the effect of each component of the system is dependent on numerous variables, such as CPU power, motherboard chipset and overall quality, total traffic load on the system, Linux kernel optimizations, network traffic, number and type of PSTN interfaces, and PSTN traffic—not to mention any non-Asterisk services the system is performing concurrently. Let's take a look at the effects of several key factors:

Codecs and transcoding
: Simply put, a *codec* (short for coder/decoder or compression/decompression) is a set of mathematical rules that define how an analog waveform will be digitized. The differences between the various codecs are due in large part to the levels of compression and quality that they offer. Generally speaking, the more compression that's required, the more work the DSP must do to code or decode the signal. Uncompressed codecs, therefore, put far less strain on the CPU (but require more network bandwidth). Codec selection must strike a balance between bandwidth and processor usage.

Central Processing Unit (and Floating Point Unit)
: A CPU is comprised of several components, one of which is the Floating Point Unit (FPU). The speed of the CPU, coupled with the efficiency of its FPU, will play a significant role in the number of users a system can effectively support. The next section, "Choosing a Processor," offers guidelines for choosing a CPU that will meet the needs of your system.

Other processes running concurrently on the system
: Being Unix-like, Linux is designed to be able to multitask several different processes. A problem arises when one of those processes (such as Asterisk) demands a very high level of responsiveness from the system. By default, Linux will equally distribute resources amongst every application that requests them. If you install a system with many different server applications, those applications will each be allowed their fair use of the CPU. Since Asterisk requires frequent high-priority access to the CPU, it does not get along well with other applications, and if Asterisk must coexist with other apps, the system may require special optimizations. This primarily involves the assignment of priorities to various applications in the system, and, during installation, careful attention to which applications are installed as services.

Kernel optimizations
: A kernel optimized for the performance of one specific application is something that very few Linux distributions offer by default, and thus it requires some thought. At the very minimum—whichever distribution you choose—a fresh copy of the Linux kernel (available from *http://www.kernel.org*) should be downloaded and compiled on your platform. You may also be able to acquire patches

that will yield performance improvements, but these are considered hacks to the officially supported kernel.

IRQ latency

Interrupt request (IRQ) latency is basically the delay between the moment a peripheral card (such as a telephone interface card) requests that the CPU stop what it's doing and the moment when the CPU actually responds and is ready to handle the task. Asterisk's peripherals (especially the Zaptel cards) are extremely intolerant of IRQ latency.

Linux has historically had problems with its ability to service IRQs quickly; this problem has caused enough trouble for audio developers that several patches have been created to address this shortcoming. So far, there has been some mild controversy over how to incorporate these patches into the Linux kernel.

Because the Digium cards require so much, it is generally recommended that only one Digium card be run in a system. If you require more connectivity than a single card can provide, either replace your existing card with one of higher density, or add another server to your environment.*

Kernel version

Asterisk is officially supported on Linux Version 2.6.

Linux distribution

Linux distributions are many and varied. In the next chapter, we will discuss the challenge of selecting a Linux distribution, and how to obtain and install both Linux and Asterisk.

Choosing a Processor

Since the performance demands of Asterisk will generally involve a large number of math calculations, it is essential that you select a processor with a powerful FPU. The signal processing that Asterisk performs can quickly demand a staggering quantity of complex mathematical computations from the CPU. The efficiency with which these tasks are carried out will be determined by the power of the FPU within that processor.

To actually name a best processor for Asterisk would fly in the face of the rapid advances in the computer industry. Even in the time between the authoring and publishing of this book, processor speeds will undergo rapid improvements, as will Asterisk's support for various architectures. Obviously, this is a good thing, but it also makes the giving of advice on the topic a thankless task. Naturally, the more

* Many people report that Sangoma cards are more robust when it comes to dealing with unpredictable motherboard chipsets, and thus can handle sharing motherboard IRQ resources. Regardless, it is still worth considering using multiple servers, as the redundancy that can be gained from this strategy can quickly offset the cost.

powerful the FPU is, the more concurrent DSP tasks Asterisk will be able to handle, so that is the ultimate consideration. When you are selecting a processor, the raw clock speed is only part of the equation. How well it handles floating-point operations will be a key differentiator, as DSP operations in Asterisk will place a large demand on it.

Both Intel and AMD CPUs have powerful FPUs. As of this writing, the Intel chips are commonly preferred for 32-bit systems, while AMD gets the nod if you're going to 64-bit. When you read this book, that may no longer be true.*

The obvious conclusion is that you should get the most powerful CPU your budget will allow. However, don't be too quick to buy the most expensive CPU out there. You'll need to keep the requirements of your system in mind—after all, a Formula 1 Ferrari is ill-suited to the rigors of rush-hour traffic.

In order to attempt to provide a frame of reference from which we can contemplate our platform decision, we have chosen to define three sizes of Asterisk systems: small, medium, and large.

Small systems

Small systems (up to 10 phones) are not immune to the performance requirements of Asterisk, but the typical load that will be placed on a smaller system will generally fall within the capabilities of a modern processor.

If you are building a small system from older components you have lying around, be aware that the resulting system cannot be expected to perform at the same level as a more powerful machine, and will run into performance degradation under a much lighter load. Hobby systems can be run successfully on very low-powered hardware,† although this is by no means recommended for anyone who is not a whiz at Linux performance tuning.

If you are setting up an Asterisk system for the purposes of learning, you will be able to build a fully featured platform using a relatively low-powered CPU. The authors run several Asterisk lab systems with 433-MHz to 700-MHz Celeron processors; the workload of these systems is typically minimal.

* If you want to be completely up to the minute on which CPUs are leading the performance race, surf on over to Tom's Hardware (*http://www.tomshardware.com*) or AnandTech (*http://www.anandtech.com*), where you will find a wealth of information about both current and out-of-date CPUs, motherboards, and chipsets.

† A 133-MHz Pentium system is known to be running Asterisk, but performance problems are likely, and properly configuring such a system requires an expert knowledge of Linux. We do not recommend running Asterisk on anything less than a 500-MHz system (for a production system, 2 GHz might be a sensible minimum), but we think the fact that Asterisk is so flexible is remarkable.

Medium systems

Medium-sized systems (from 10 to 50 phones) are where performance considerations will be the most challenging to resolve. Generally, these systems will be deployed on one or two servers only, and thus each machine will be required to handle more than one specific task. As loads increase, the limits of the platform will become increasingly stressed. Users may begin to perceive quality problems without realizing that the system is not faulty in any way, but simply exceeding its capacity. These problems will get progressively worse as more and more load is placed on the system, with the user experience degrading accordingly. It is critical that performance problems be identified and addressed before they are noticed by users.

Monitoring performance on these systems and quickly acting on any developing trends is a key to ensuring that a quality telephony platform is provided.

Large systems

Large systems (over 50 users) can be distributed across multiple cores, and thus performance concerns can be managed through the addition of machines. Very large Asterisk systems—from 500 to over 1,000 users—have been created in this way. Building a large system requires an advanced level of knowledge in many different disciplines. We will not discuss it in detail in this book, other than to say that the issues you'll encounter will be similar to those encountered during any deployment of multiple servers handling a single, distributed task.

Choosing a Motherboard

Just to get any anticipation out of the way, we also cannot recommend specific motherboards in this book. With new motherboards coming out on a weekly basis, any recommendations we made would be rendered moot by obsolescence before the published copy hit the shelves. Not only that, but motherboards are like automobiles: while they are all very similar in principle, the difference is in the details. And as Asterisk is a performance application, the details matter.

What we will do, therefore, is give you some idea of the kinds of motherboards that can be expected to work well with Asterisk, and the features that will make for a good motherboard. The key is to have both stability and high performance. Here are some guidelines to follow:

- The various system buses must provide the minimum possible latency. If you are planning a PSTN connection using analog or PRI interfaces (discussed later in this chapter), the Digium Zaptel cards will generate 1,000 interrupt requests per second. Having devices on the bus that interfere with this process will result in degradation of call quality. Chipsets from Intel (for Intel CPUs) and nVidia nForce (for AMD CPUs) seem to score the best marks in this area. Review the specific chipset of any motherboard you are evaluating to ensure that it does not have known problems with IRQ latency.

- If you are running Zaptel cards in your system, you will want to ensure that your BIOS allows you maximum control over IRQ assignment. As a rule, high-end motherboards will offer far greater flexibility with respect to BIOS tweaking; value-priced boards will generally offer very little control. This may be a moot point, however, as APIC-enabled motherboards turn IRQ control over to the operating system.
- Server-class motherboards generally implement a different PCI standard than workstation-class motherboards. While there are many differences, the most obvious and well known is that the two versions have different voltages. Depending on which cards you purchase, you will need to know if you require 3.3V or 5V PCI slots. Figure 2-1 shows the difference between 3.3V and 5V slots. Most server motherboards will have both types, but workstations will typically have only the 5V version.

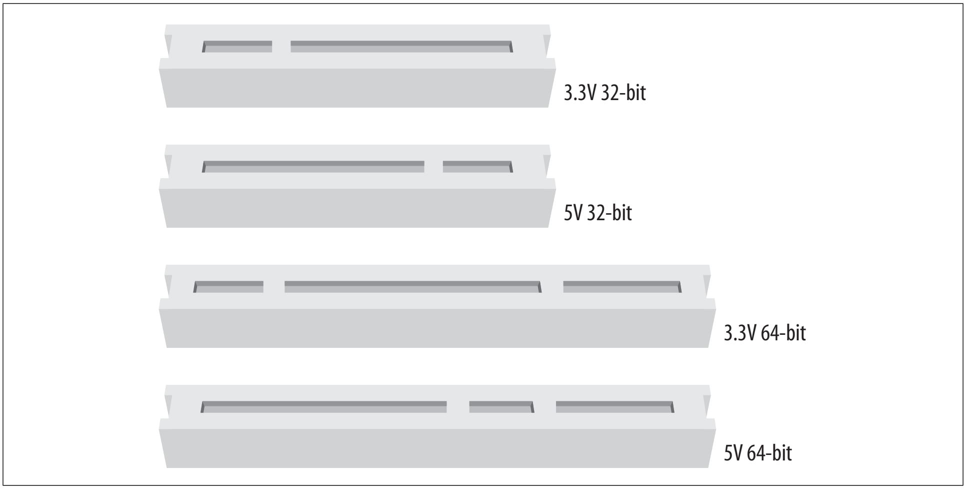

Figure 2-1. Visual identification of PCI slots

- Consider using multiple processors. This will provide an improvement in the system's ability to handle multiple tasks. For Asterisk, this will be of special benefit in the area of floating-point operations.

It should be noted that evidence now suggests that connecting together two completely separate, single-CPU systems may provide far more benefits than simply using two processors in the same machine. You not only double your CPU power, but you also achieve a much better level of redundancy at a similar cost to a single-chassis, dual-CPU machine. Keep in mind, though, that a dual-server Asterisk solution will be more complex to design than a single-machine solution.

- Avoid motherboards that include built-in audio and video components. If you want a sound card, install one. As for a video card, you may not need one at all—certainly Asterisk does not require one. It has traditionally been the more value-priced motherboards that have had these components on-board, and these board designs often make compromises to keep down the costs.
- If possible, install an external modem. If you must have an internal modem, you will need to ensure that it is not a so-called "Win-modem"—it must be a completely self-sufficient unit (note that these are very difficult, if not impossible, to find).
- Consider that with built-in networking, if you have a network component failure, the entire motherboard will need to be replaced. On the other hand, if you install a peripheral Network Interface Card (NIC), there may be an increased chance of failure due to the extra mechanical connections involved. Some value can probably be gained from having both primary and backup cards installed in the system.
- The stability and quality of your Asterisk system will be dependent on the components you select for its architecture. Asterisk is a beast, and it expects to be fed the best. As with just about anything, high cost is not always synonymous with quality, but you will want to become a connoisseur of computer components.

Having said all that, we need to get back to the original point: Asterisk can and will happily install on pretty nearly any PC platform. The lab systems used to write this book, for example, included everything from a 433-MHz Pentium III on an Intel chipset to an Athlon XP 2000 on a VIA-based motherboard. We have not experienced any performance or stability problems running less than five concurrent telephone connections. For the purposes of learning, do not be afraid to install Asterisk on whatever system you can scrounge up. When you are ready to put your system into production, however, you will need to understand the ramifications of the choices you make with respect to your hardware.

Power Supply Requirements

One often-overlooked component in a PC is the power supply (and the supply of power). For a telecommunications system, these components can play a significant role in the quality of the user experience.

Computer power supplies

The power supply you select for your system will play a vital role in the stability of the entire platform. Asterisk is not a particularly power-hungry application, but anything relating to multimedia (whether it be telephony, professional audio, video, or the like) is generally sensitive to power *quality*.

This oft-neglected component can turn an otherwise top-quality system into a poor performer. By the same token, a top-notch power supply might enable an otherwise cheap PC to perform like a champ.

The power supplied to a system must provide not only the energy the system needs to perform its tasks, but also stable, clean signal lines for all of the voltages your system expects from it.

Redundant power supplies

In a carrier-grade or high-availability environment, it is common to deploy servers that use a redundant power supply. Essentially, this involves two completely independent power supplies, either one of which is capable of supplying the power requirements of the system.

If this is important to you, keep in mind that best practices suggest that to be properly redundant, these power supplies should be connected to completely independent Uninterruptible Power Supplies (UPSs) that are in turn fed by totally isolated electrical circuits.

Environment

Your system's environment consists of all those factors that are not actually part of the server itself, but nevertheless play a crucial role in the reliability and quality that can be expected from the system. Electrical supplies, room temperature and humidity, sources of interference, and security are all factors that should be contemplated.

Power Conditioning and Uninterruptible Power Supplies

When selecting the power sources for your system, consideration should be given not only to the amount of power the system will use, but also to the manner in which that power is delivered.

Power is not as simple as voltage coming from the outlet in the wall, and you should never just plug a production system into whatever electrical source is near at hand.[*] Giving some consideration to the supply of power to your system can provide a far more stable power environment, leading to a far more stable system.

[*] Okay, look, you *can* plug it in wherever you'd like, and it'll probably work, but if your system has strange stability problems, please give this section another read. Deal?

Properly grounded, conditioned power feeding a premium-quality power supply will ensure a clean *logic ground* (a.k.a. 0-volt) reference* for the system and keep electrical noise on the motherboard to a minimum. These are industry-standard best practices for this type of equipment, which should not be neglected. A relatively simple way to achieve this is through the use of a *power-conditioned* UPS.†

Power-conditioned UPSs

The UPS is well known for its role as a battery backup, but the power-conditioning benefits that high-end UPS units also provide are less well understood.

Power conditioning can provide a valuable level of protection from the electrical environment by regenerating clean power through an isolation transformer. A quality power conditioner in your UPS will eliminate most electrical noise from the power feed and help to ensure a rock-steady supply of power to your system.

Unfortunately, not all UPS units are created equal; many of the less expensive units do not provide clean power. What's worse, manufacturers of these devices will often promise all kinds of protection from surges, spikes, overvoltages, and transients. While such devices may protect your system from getting fried in an electrical storm, they will not clean up the power being fed to your system, and thus will do nothing to contribute to stability.

Make sure your UPS is power conditioned. If it doesn't say exactly that, it isn't.

Grounding

Voltage is defined as the difference in electrical potential between two points. When considering a *ground* (which is basically nothing more than an electrical path to earth), the common assumption is that it represents 0 volts. But if we do not define that 0V in *relation* to something, we are in danger of assuming things that may not be so. If you measure the voltage between two grounding references, you'll often find that there is a voltage potential between them. This voltage potential between grounding points can be significant enough to cause logic errors—or even damage—in a system where more than one path to ground is present.

* In electronic devices, a binary zero (0) is generally related to a 0-volt signal, while a binary one (1) can be represented by many different voltages (commonly between 2.5 and 5 volts). The grounding reference that the system will consider 0 volts is often referred to as the "logic ground." A poorly grounded system might have electrical potential on the logic ground to such a degree that the electronics mistake a binary zero for a binary one. This can wreak havoc with the system's ability to process instructions.

† It is a commonly misunderstood belief that all UPSs provide clean power. This is not at all true.

One of the authors recalls once frying a sound card he was trying to connect to a friend's stereo system—even though both the computer and the stereo were in the same room, more than 6 volts of difference was measured between the ground conductors of the two electrical outlets they were plugged into! The wire between the stereo and the PC (by way of the sound card) provided a path that the voltage eagerly followed, thus frying a sound card that was not expecting an electrical current on its signal leads. Connecting both the PC and the stereo to the same outlet fixed the problem.

When considering electrical regulations, the purpose of a ground is primarily human safety. In a computer, the ground is used as a 0V logic reference. An electrical system that provides proper safety will not always provide a proper logic reference—in fact, the goals of safety and power quality are sometimes in disagreement. Naturally, when a choice must be made, safety has to take precedence.

Since the difference between a binary zero and a binary one is represented in computers by voltage differences of sometimes less than 3V, it is entirely possible for unstable power conditions caused by poor grounding or electrical noise to cause all manner of intermittent system problems. Some power and grounding advocates estimate that more than 80% of unexplained computer glitches can be traced to power quality.

Modern switching power supplies are somewhat isolated from power quality issues, but any high-performance system will always benefit from a well-designed power environment. In mainframes, proprietary PBXs, and other expensive computing platforms, the grounding of the system is never left to chance. The electronics and frames of these systems are always provided with a dedicated ground that does not depend on the safety grounds supplied with the electrical feed.

Regardless of how much you are willing to invest in grounding, when you specify the electrical supply to any PBX, ensure that the electrical circuit is completely dedicated to your system (as discussed in the next section) and that an insulated, isolated grounding conductor is provided. This can be expensive to provision, but it will contribute greatly to a quality power environment for your system.[*]

It is also vital that each and every peripheral you connect to your system be connected to the same electrical receptacle (or, more specifically, the same ground reference). This will cut down on the occurrence of ground loops, which can cause anything from buzzing and humming noises to damaged or destroyed equipment.

[*] On a hobby system, this is probably too much to ask, but if you are planning on using Asterisk for anything important, at least be sure to give it a fighting chance—don't put anything like air conditioners, photocopiers, laser printers, or motors on the same circuit.

Electrical Circuits

If you've ever seen the lights dim when an electrical appliance kicks in, you've seen the effect that a high-energy device can have on an electrical circuit. If you were to look at the effects of a multitude of such devices, each drawing power in its own way, you would see that the harmonically perfect 50- or 60-Hz sine wave you may think you're getting with your power is anything but. Harmonic noise is extremely common on electrical circuits, and it can wreak havoc on sensitive electronic equipment. For a PBX, these problems can manifest as audio problems, logic errors, and system instability.

Never install a server on an electrical circuit that is shared with any other devices. There should be only one outlet on the circuit, and you should connect only your telephone system (and associated peripherals) to it. The wire (including the ground) should be run unbroken directly back to the electrical panel. The grounding conductor should be insulated, and isolated. There are far too many stories of photocopiers, air conditioners, and vacuum cleaners wreaking havoc with sensitive electronics to ignore this rule of thumb.

The electrical regulations in your area must always take precedence over any ideas presented here. If in doubt, consult a power quality expert in your area on how to ensure that you adhere to electrical regulations. Remember, electrical regulations take into account the fact that human safety is far more important than the safety of the equipment.

The Equipment Room

Environmental conditions can wreak havoc on systems, and yet it is quite common to see critical systems deployed with little or no attention given to these matters. If one looks at the statistics, it becomes obvious that attention to environmental factors can play a significant role in the stability and reliability of systems.

Humidity

Simply put, humidity is water in the air. Water is a disaster for electronics, for two main reasons: 1) water is a catalyst for corrosion, and 2) water is conductive enough that it can cause short circuits. Do not install any electronic equipment in areas of high humidity, without providing a means to remove the moisture.

Temperature

Heat is the enemy of electronics. The cooler you keep your system, the more reliably it will perform. If you cannot provide a properly cooled room for your system, at a minimum ensure that it is placed in a location that ensures a steady supply of clean, cool air. Also, keep the temperature steady. Changes in temperature can lead to condensation and other damaging changes.

Dust

There is an old adage in the computer industry that holds that dust bunnies inside of a computer are lucky. Let's consider some of the realities of dust bunnies:

- Significant buildup of dust can restrict airflow inside the system, leading to increased levels of heat.
- Dust can contain metal particles, which, in sufficient quantities, can contribute to signal degradation or shorts on circuit boards.

Put critical servers in a filtered environment, and clean out dust bunnies on a regular schedule.

Security

Server security naturally involves protecting against network-originated intrusions, but the environment also plays a part in the security of a system. Telephone equipment should always be locked away, and only persons who have a need to access the equipment should be allowed near it.

Telephony Hardware

If you are going to connect Asterisk to any legacy telecommunications equipment, you will need the correct hardware. The hardware you require will be determined by what it is you want to achieve.

Connecting to the PSTN

Asterisk allows you to seamlessly bridge circuit-switched telecommunications networks[*] with packet-switched data networks.[†] Because of Asterisk's open architecture (and open source code), it is ultimately possible to connect any standards-compliant interface hardware. The selection of open source telephony interface boards is currently limited, but as interest in Asterisk grows, that will rapidly change.[‡] At the moment, one of the most popular and cost-effective ways to connect to the PSTN is

[*] Often referred to as *TDM networks*, due to the Time Division Multiplexing used to carry traffic through the PSTN.

[†] Popularly called VoIP networks, although Voice over IP is not the only method of transmitting voice over packet networks (Voice over Frame Relay was very popular in the late 1990s).

[‡] The evolution of inexpensive, commodity-based telephony hardware is only slightly behind the telephony software revolution. New companies spring up on a weekly basis, each one bringing new and inexpensive standards-based devices into the market.

to use the interface cards that evolved from the work of the Zapata Telephony Project (*http://www.zapatatelephony.org*).

Analog interface cards

Unless you need a lot of channels (or a have lot of money to spend each month on telecommunications facilities), chances are that your PSTN interface will consist of one or more analog circuits, each of which will require a Foreign eXchange Office (FXO) port.

Digium, the company that sponsors Asterisk development, produces the most popular analog interface card for Asterisk, known as the TDM400P.* The TDM400P is a four-port base card that allows for the insertion of up to four daughter cards, which deliver either FXO or Foreign eXchange Station (FXS) ports. The TDM400P can be purchased with these cards preinstalled, and Digium has designated part numbers to describe these configurations. The naming convention is TDM *x y* B, where *x* and *y* are numbers representing the quantity of FXS and FXO† cards on the board, respectively. Check out Digium's web site (*http://www.digium.com*) for more information about this card.

An older card produced by Digium was known as the X100P. It is no longer available from Digium, but you may be able to find a clone of this card.

Another company that produces Asterisk-compatible analog cards is Voicetronix. They have three Asterisk cards in their analog lineup: OpenLine4, OpenSwitch6, and OpenSwitch12.

Digital interface cards

If you require more than 10 circuits, or require digital connectivity, chances are you're going to be in the market for a T1 or E1 card.‡ Bear in mind, though, that the monthly charges for a digital PSTN circuit vary widely. In some places, as few as five circuits can justify a digital circuit; in others, the technology may never be cost-justifiable. The more competition there is in your area, the better chance you have of finding a good deal. Be sure to shop around.

The Zapata Telephony Project originally produced a T1 card, the Tormenta, that is the ancestor of most Asterisk-compatible T1 cards. The original Tormenta cards are now considered obsolete, but they do still work with Asterisk. Currently, the only company known to be producing these cards is Varion.

* The TDM400P is not, in fact, a TDM card at all. It is analog.

† FXS and FXO refer to the opposing ends of an analog circuit. Which one you need will be determined by what you want to connect to. Chapter 7 discusses these in more detail.

‡ T1 and E1 are digital telephony circuits. We'll discuss them further in Chapter 7.

Digium makes several different digital circuit interface cards. The features on the cards are the same; the primary differences are whether they provide T1 or E1 interfaces, and how many interfaces each card provides. Although it's technically possible, the general consensus in the Asterisk community is that no more than one of these cards should be deployed in a single system.

Sangoma, who have been producing open source WAN cards for many years, have recently added Asterisk support for their T1/E1 cards.[*] Sangoma's cards contain powerful field-programmable gate arrays (FPGAs), which make them extremely flexible. In an Asterisk environment, for example, they have been programmed to interface with the Zapata channel driver.

Channel banks

A *channel bank* is loosely defined as a device that allows a digital circuit to be demultiplexed into several analog circuits (and vice versa). More specifically, a channel bank lets you connect analog telephones and lines into a system across a T1 line. Figure 2-2 shows how a channel bank fits into a typical office phone system.

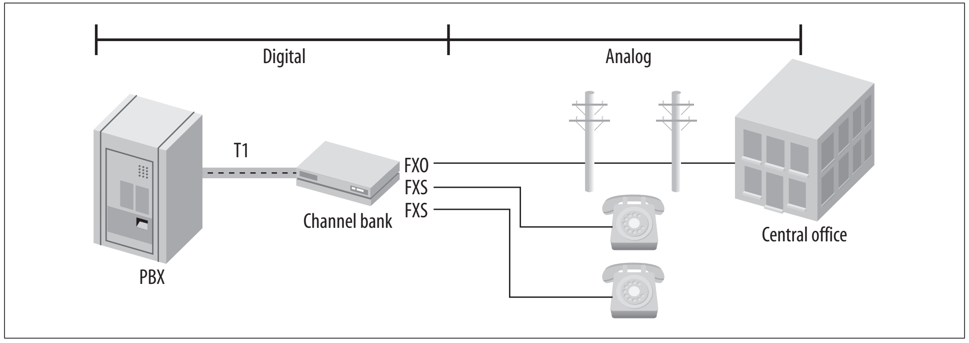

Figure 2-2. Channel bank

Although they can be expensive to purchase, many people feel very strongly that the only proper way to integrate analog circuits and devices into Asterisk is through a channel bank.

Other types of PSTN interfaces

Many VoIP gateways exist that can be configured to provide access to PSTN circuits. Generally speaking, these will be of most use in a smaller system (one or two lines). They can also be very complicated to configure, as the interaction between the various networks and devices requires a solid understanding of both telephony and VoIP

[*] It should be noted that a Sangoma Frame Relay card figured prominently in the original development of Asterisk (see *http://linuxdevices.com/articles/AT8678310302.html*)—Sangoma has a long history of supporting open source WAN interfaces with Linux.

fundamentals. For that reason, we will not discuss these devices in detail in this book. They are worth looking into, however—popular units are made by Sipura, Grandstream, Digium, and many other companies.

Another way to connect to the PSTN is through the use of Basic Rate Interface (BRI) ISDN circuits. BRI is a digital telecom standard that specifies a two-channel circuit that can carry up to 144 kbps of traffic. It is very rarely used in North America and most of the rest of the world, but it's quite popular in Europe. Due to the variety of different ways this technology has been implemented, we will not be discussing BRI in very much detail in this book.

Connecting Exclusively to a Packet-Based Telephone Network

If you do not need to connect to the PSTN, Asterisk requires no hardware other than a server with a Network Interface Card.

However, if you are going to be providing music on hold or conferencing and you have no physical timing source, you will need the *ztdummy* Linux kernel module. *ztdummy* is a clocking mechanism designed to provide a timing source to a system where no hardware timing source exists. In Version 2.4 of the Linux kernel, to use *ztdummy* you must have a UHCI-type USB controller on your motherboard. In Linux 2.6, that requirement is no more.

Types of Phone

Since the title of this book is *Asterisk: The Future of Telephony*, we would be remiss if we didn't discuss the devices that all of this technology ultimately has to interconnect: telephones!

We all know what a telephone is—but will it be the same five years from now? Part of the revolution that Asterisk is contributing to is the evolution of the telephone, from a simple audio communications device into a multimedia communications terminal providing all kinds of yet-to-be-imagined functions.

As an introduction to this exciting concept, we will briefly discuss the various kinds of devices we currently call "telephones" (any of which can easily be integrated with Asterisk). We will also discuss some ideas about what these devices may evolve into in the future (devices that will also easily integrate with Asterisk).

Physical Telephones

Any physical device whose primary purpose is terminating an on-demand audio communications circuit between two points can be classified as a physical telephone. At a minimum, such a device has a handset and a dial pad; it may also have feature keys, a display screen, and various audio interfaces.

This section takes a brief look at the various user (or endpoint) devices you might want to connect to your Asterisk system. We'll delve more deeply into the mechanics of analog and digital telephony in Chapter 7.

Analog telephones

Analog phones have been around since the invention of the telephone. Up until about 20 years ago, all telephones were analog. Although analog phones have some technical differences in different countries, they all operate on similar principles.

When a human being speaks, the vocal cords, tongue, teeth, and lips create a complex variety of sounds. The purpose of the telephone is to capture these sounds and convert them into a format suitable for transmission over wires. In an analog telephone, the transmitted signal is *analogous* to the sound waves produced by the person speaking. If you could see the sound waves passing from the mouth to the microphone, they would be proportional to the electrical signal you could measure on the wire.

> This contiguous connection is referred to as a *circuit*, which the telephone network used to use electromechanical switches to create; hence the term *circuit-switched network*.

Analog telephones are the only kind of phone that are commonly available in any retail electronics store. In the next few years, that can be expected to change dramatically.

Proprietary digital telephones

As digital switching systems developed in the 1980s and 1990s, telecommunications companies developed digital Private Branch eXchanges (PBXs) and Key Telephone Systems (KTSs). The proprietary telephones developed for these systems were completely dependent on the systems to which they were connected and could not be used on any other systems. Even phones produced by the same manufacturer were not cross-compatible (for example, a Nortel Norstar set will not work on a Nortel Meridian 1 PBX). The proprietary nature of digital telephones limits their future. In this emerging era of standards-based communications, they will quickly be relegated to the dustbin of history.

The handset in a digital telephone is generally identical in function to the handset in an analog telephone, and they are often compatible with each other. Where the digital phone is different is that inside the telephone, the analog signal is sampled and converted into a digital signal—that is, a numerical representation of the analog waveform. We'll leave a detailed discussion of digital signals until Chapter 7; for now, suffice it to say that the primary advantage of a digital signal is that it can be transmitted over limitless distances with no loss of signal quality.

The chances of anyone ever making a proprietary digital phone directly compatible with Asterisk are fairly small, but companies such as Citel (*http://www.citel.com*) have created gateways that convert the proprietary signals to SIP.[*]

ISDN telephones

Prior to VoIP, the closest thing to a standards-based digital telephone was an ISDN-BRI terminal. Developed in the early 1980s, ISDN was expected to revolutionize the telecommunications industry in exactly the same way that VoIP promises to finally achieve today.

> There are two types of ISDN: *Primary Rate Interface* (PRI) and *Basic Rate Interface* (BRI). PRI is commonly used to provide trunking facilities between PBXs and the PSTN, and is widely deployed. BRI is not at all common in North America, but has enjoyed some success in Europe.

While ISDN was widely deployed by the telephone companies, many consider the standard to have been a flop, as it generally failed to live up to its promises. The high costs of implementation, recurring charges, and lack of cooperation amongst the major players contributed to an environment that caused more problems than it solved.

BRI was intended to service terminal devices and smaller sites (a BRI loop provides two digital circuits). While a wealth of BRI devices have been developed, BRI has largely been deprecated in favor of faster, less expensive technologies such as ADSL, cable modems, and VoIP.

BRI is still very popular for use in video-conferencing equipment, as it provides a fixed bandwidth link. Also, BRI does not have the type of quality of service issues a VoIP connection might, as it is circuit-switched.

BRI is still sometimes used in place of analog circuits to provide trunking. Whether or not this is a good idea depends mostly on how your local phone company prices the service, and what features it is willing to provide.

IP telephones

IP telephones are heralds of the most exciting change in the telecommunications industry. In the very near future, standards-based IP telephones will be available in retail stores.[†] The wealth of possibilities inherent in these devices will cause an explosion of interesting applications, from video phones, to high-fidelity broadcasting devices, to wireless mobility solutions, to purpose-built sets for particular industries, to flexible all-in-one multimedia systems.

[*] The Session Initiation Protocol is currently the most well-known and popular protocol for VoIP. We will discuss it further in Chapter 8.

[†] As of this writing, Wal-Mart was offering a basic IP telephone on its web site (*http://www.walmart.com*).

The revolution that IP telephones will spawn has nothing to do with a new type of wire to connect your phone to, and everything to do with giving you the power to communicate the way you want.

The early-model IP phones that have been available for several years now do not represent the future of these exciting appliances. They are merely a stepping-stone; a familiar package in which to wrap a fantastic new way of thinking.

The future is far more promising.

Soft Phones

A *soft phone* is a software program that provides telephone functionality on a non-telephone device, such as a PC or PDA. So how do we recognize such a beast? What might at first glance seem a simple question actually raises many. A soft phone should probably have some sort of dial pad, and it should provide an interface that reminds users of a telephone. But will this always be the case?

The term "soft phone" can be expected to evolve rapidly, as our concept of what exactly a telephone is undergoes a revolutionary metamorphosis. As an example of this evolution, consider the following: would we correctly define popular communication programs such as Instant Messenger as soft phones? IM provides the ability to initiate and receive standards-based VoIP connections. Does this not qualify it as a soft phone? Answering that question requires knowledge of the future that we do not yet possess. Suffice it to say that while at this point in time, soft phones are expected to look and sound like traditional phones, that conception is likely to change in the very near future.

As standards evolve and we move away from the traditional telephone and toward a multimedia communications culture, the line between soft phones and physical telephones will become blurred indeed. For example, we might purchase a communications terminal to serve as a telephone, and install a soft phone program onto it to provide the functions we desire.

Having thus muddied the waters, the best we can do at this point is to define what the term "soft phone" will refer to in relation to this book, with the understanding that the meaning of the term can be expected to undergo a massive change over the next few years. For our purposes, we will define a soft phone: any device that runs on a personal computer, presents the look and feel of a telephone, and provides as its primary function the ability to make and receive full-duplex audio communications (formerly known as "phone calls")[*] through E.164 addressing.[†]

[*] OK, so you think you know what a phone call is? So did we. Let's just wait a few years, shall we?

[†] E.164 is the ITU standard that defines how phone numbers are assigned. If you've used a telephone, you've used E.164 addressing.

Telephony Adaptors

A *telephony adaptor* (usually referred to as an ATA, or Analog Terminal Adaptor) can loosely be described as an end-user device that converts communications circuits from one protocol to another. Most commonly, these devices are used to convert from some digital (IP or proprietary) signal to an analog connection that you can plug a standard telephone or fax machine into.

These adaptors could be described as gateways, for that is their function. However, popular usage of the term *telephony gateway* would probably best describe a multiport telephony adaptor, generally with more complicated routing functions.

Telephony adaptors will be with us for as long as there is a need to connect incompatible standards and old devices to new networks. Eventually, our reliance on these devices will disappear, as did our reliance on the modem—obsolescence through irrelevance.

Communications Terminals

Communications terminal is an old term that disappeared for a decade or two and is being reintroduced here, very possibly for no other reason than that it needs to be discussed so that it can eventually disappear again—once it becomes ubiquitous.

First, a little history. When digital PBX systems were first released, manufacturers of these machines realized that they could not refer to their endpoints as telephones—their proprietary nature prevented them from connecting to the PSTN. They were therefore called *terminals*, or stations. Users, of course, weren't having any of it. It looked like a telephone and acted like a telephone, and therefore it *was* a telephone. You will still occasionally find PBX sets referred to as terminals, but for the most part they are called telephones.

The renewed relevance of the term "communications terminal" has nothing to do with anything proprietary—rather, it's the opposite. As we develop more creative ways of communicating with each other, we gain access to many different devices that will allow us to connect. Consider the following scenarios:

- If I use my PDA to connect to my voicemail and retrieve my voice messages (converted to text), does my PDA become a phone?
- If I attach a video camera to my PC, connect to a company's web site, and request a live chat with a customer service rep, is my PC now a telephone?
- If I use the IP phone in my kitchen to surf for recipes, is that a phone call?

The point is simply this: we'll probably always be "phoning" each other, but will we always be using "telephones" to do so?

Linux Considerations

If you ask anyone at the Free Software Foundation, they will tell you that what we know as Linux is in fact GNU/Linux. All etymological arguments aside, there is some valuable truth to this statement. While the kernel of the operating system is indeed Linux, the vast majority of the utilities installed on a Linux system and used regularly are in fact GNU utilities. “Linux” is probably only 5% Linux, possibly 75% GNU, and perhaps 20% everything else.

Why does this matter? Well, the flexibility of Linux is both a blessing and a curse. It is a blessing because with Linux you can truly craft your very own operating system from scratch. Since very few people ever do this, the curse is in large part due to the responsibility you must bear in determining which of the GNU utilities to install, and how to configure the system.

If this seems overwhelming, do not fear. In the next chapter, we will discuss the selection, installation, and configuration of the software environment for your Asterisk system.

Conclusion

In this chapter, we’ve discussed all manner of issues that can contribute to the stability and quality of an Asterisk installation. Before we scare you off, we should tell you that many people have installed Asterisk on top of a graphical Linux workstation, running a web server, a database, an X-windowing environment, and who knows what else, with no problems whatsoever. How much time and effort you should devote to following the best practices and engineering tips in this chapter all depends on how much work you expect the Asterisk server to perform, and how much quality and reliability your system must provide.

What we have attempted to do in this chapter is give you a feel for the kinds of best practices that will help to ensure that your Asterisk system will be built on a reliable, stable platform. Asterisk is quite willing to operate under far worse conditions, but the amount of effort and consideration you decide to give these matters will play a part in the stability of your PBX. Your decision should depend on how critical your Asterisk system will be.