# The Role of SMB in Embedded Devices

## Introduction

This article is the first of two relating to the use of the SMB network file-sharing stack on embedded systems. A second article will discuss considerations that a development team may wish to address when choosing an embedded SMB stack.

The article will first describe what an embedded device is from the perspective of an SMB stack and then give some detail on what an embedded SMB stack is. It will also cover use cases for embedded SMB stacks.

Connected Way (https://www.connectedway.com) has been developing our embedded SMB stack for over 20 years. The perspectives and considerations I share in this document are the results of our experience in this market.

The article series is intended to be beneficial to product managers, development managers, and developers of embedded products that wish to consider the integration of SMB network file system features in their products.

## What is an Embedded Device?

There was a time when this was an easy question to answer. Embedded devices ran a scaled-down operating system or no operating system at all. They ran on less powerful CPUs and had limited resources at their disposal. Probably the most differentiating factor is that they did not have storage.

This is no longer the case. Many embedded devices run the same operating system as run on Desktop and server computers. They run on CPUs just as powerful if not greater than multi-purpose computers. Flash memory and Solid-State Disks (SSDs) are prevalent on embedded devices. So, from a hardware and software perspective, an embedded device may be no different from the hardware and software running on a typical desktop.

The differentiation is in how the device is used. An embedded device is purpose-built, meaning the device was designed to serve a specific purpose. It may be designed to navigate, route packets over a network, or control a piece of machinery. Although perhaps capable of performing other tasks, its form factor, packaging, and interfaces make it most appropriate for its designed purpose.

For example, a Linksys WRT3200 (see Figure 1) has a multicore ARMv7 processor with 512 MB of storage and it runs Linux. Although perhaps not quite as powerful as today's desktop computers, it could run simple games or a database. Yet it is embedded because it has been designed as a router.


*Figure 1. Image of a Linksys WRT3200 Router (Source: https://openwrt.org/toh/linksys/wrt3200acm)*

The Linksys router is an example of an embedded device that runs open-source software. That is a device that runs software that is free, and available to modify, build, and install on the device. Not all embedded devices are as open. Most embedded devices have their operational software built and installed in the factory, and although the device may be upgradeable, it is only upgraded using validated upgrade media.

## What does Real-Time Embedded mean?

Somewhat synonymous with embedded, so much so that the terms are often used interchangeably, is the notion of real-time systems. Embedded systems historically used what was referred to as "Real-Time Operating Systems" (RTOS). Linux has a tuning referred to as "Real-Time Linux". You'll hear of their usefulness in relation to the ability to operate in real time. For example, a car's ABS braking system is managed by a real-time computer. When I step on the brakes, I want the brakes to operate in real-time, as in, right now!

But real-time is essentially a notion of efficiency. If I have a computer that is infinitely fast but processes everything in batch mode, it wouldn't really matter whether the system was specifically designed for real-time because I will meet my time requirements regardless of the design of the operating system and the software.

Yet our computers are not infinitely fast. And embedded systems, being purpose-built, are designed with only enough horsepower to handle the requirements of the product. The more efficient we can design our software to be, the more we can optimize the hardware design and minimize costs. For that reason, embedded and real-time have become synonymous.

Often, another term will get thrown into the mix: "Event-Driven". "Real-Time Event-Driven Embedded System". Event-driven is a technique to achieve real-time. This will be discussed more when I talk about scalability.

## What is SMB?

SMB, or Server Message Block, is a file-sharing protocol initially used by Microsoft operating systems. A file-sharing protocol defines a set of network packets that effectively share the contents of files on one system with another system and allow the implementation of a network file system.

It's worth noting that there is a difference between file sharing and file transfer although the distinction is not universally shared. File transfer allows the transfer of a file as a complete entity. Examples of file transfer protocols are HTTP/S, FTP, TFTP, and SSH/SCP. On the other hand, file sharing allows a file on one system to be opened and read or written as needed by another system. Only those portions of files of interest to the applications need to be communicated over the network.

A file system is generally a mechanism to arrange, store, assign metadata to content, and look up content. Most computer literate people are familiar with the structure of a file system. A typical example is shown in Figure 2.

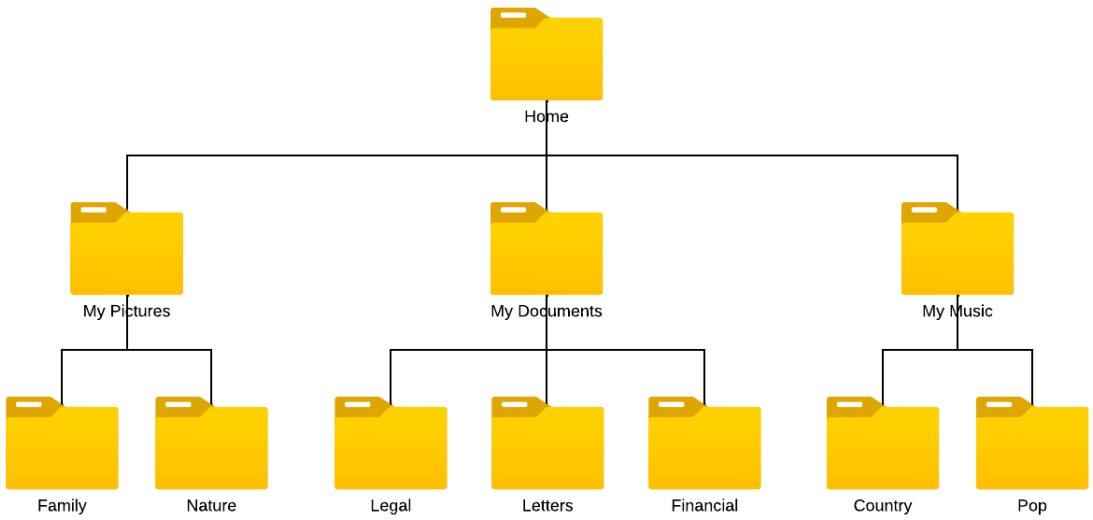


*Figure 2. Sample File System Structure*

This abstraction of a file system that exists on a storage device local to a system is the same as that for a network file system. Common network file systems are NFS, AFP, and SMB.

Therefore, SMB is considered both the protocol that shares files and the abstraction of the server's file systems to the client. It provides authentication of users, synchronization primitives to ensure the integrity of the content, confidentiality of the content as needed while in transit, remote file system maintenance, and a lot of other capabilities.

Technology provides a lot of alternatives. Rarely is there only one way to accomplish some goal but rather there are many ways, each with its own pros and cons. I discussed the difference between file transfer and file sharing, but you can usually access content either way. If I want to read a random block in a file, I could transfer the entire file to my local system, and then access the block I'm

interested in locally. To update a file, I could copy the file locally, change it, and write it back. I'd just want to be sure that the original copy hasn't changed before I commit my changes. Otherwise, I may lose some data along the way. If I want to make a copy of a file, I could transfer it using FTP, or I could use file sharing and open it, then sequentially read each block while writing to a copy. What mechanism is more natural to your application and environment?

A network file system provides a natural way to organize content. File systems are typically made up of volumes with a hierarchy of directories or folders populated with content. It is straightforward to browse this hierarchy to locate desired content or to find a place to store new content. One could argue that you don't need a network file system like SMB to accomplish this. FTP provides a mechanism to navigate around a directory structure and obtain listings. There are plug-ins to file explorer applications that can abstract a remote file system using FTP. What is the best approach will always depend on the design of the application.

File system abstractions have also been built for web browsers. These abstractions allow a user to navigate around a remote file system and browse and manage content with ease. There are online cloud infrastructures like Google Drive, One Drive, Apple Drive, and many private clouds. Why use an SMB-based network file system rather than one of these services? Once again, it depends.

Although there are many considerations in choosing the type of network file system, the primary considerations are three:

- How is the content accessed? Randomly? Sequentially? Synchronously? Asynchronously? Sparse?
- How is the content synchronized? Does it require record or file locking? Is it transaction based? Does it require access and sharing controls?
- How is the content stored and shared? Does the content reside on an existing file system? Are you sharing individual files or entire directories?

That last bullet about how the content is stored and shared is significant. All of today's desktop and server operating systems have built-in support to share content using SMB. SMB exports content directly from the host's file system without the need for any additional software. SMB obeys synchronization, sharing, and access rules without the need to define new access lists. Other than registering shared directories, content does not need to be explicitly added to the exported directories to be shared. Any local changes to the content are automatically and immediately reflected in the exported content. SMB exports are the easiest and most powerful sharing mechanism available.

## What are the use cases for SMB in embedded devices?

This brings us to embedded SMB, the topic of this article. Why would you want to add a network file system to an embedded device? What capabilities would it provide? How would adding a network file system to your device differentiate it from competing devices?

Obviously not every embedded device benefits from a network file system. The Linksys router I discussed earlier has little use for a network file system, at least for its primary purpose of routing network packets. But that's not to say that supporting a network file system in the device is not useful.

I've divided the market for technology devices into five categories. There are many ways to look at the markets and there is no definitive set, but this is a useful set for this article:

- Consumer and Office Electronics
- Communications
- Healthcare
- Industrial Controls
- Other Markets

## Consumer and Office Electronics Use Cases

This is the biggest market for embedded SMB perhaps because of the desire to differentiate and add value to a manufacturer's products.

- Audio/Video Players. A/V players play content. Content is typically represented as files. Even if the meta-data associated with the content is stored in some database, the content itself is stored in a file within some file system. In fact, most Blue-Ray players today are capable of streaming content from PCs or storage devices using SMB.
- Cameras. Cameras are designed to record content. As with A/V players, the content is stored initially in the camera as a collection of files. The content can be transferred to PCs and NAS devices by moving around SD cards, but many cameras are able to transfer content to the desired target by using SMB.
- Printers. Some Wi-Fi-enabled printers accept content to be printed using the SMB protocol. In this case, the printer would implement a type of SMB server.
- Scanners. Standalone scanners and scanners as part of a multi-function printer are one of the most widely deployed use cases for SMB. Scanners typically allow a user to transfer the scanned document to a destination either as part of an email message or the image can be stored directly on a target computer using a "Scan to Computer" function in the device.

## Communication Device Use-Case

Shared storage is increasingly becoming a standard feature in most networks. There are many manufacturers of Network Attached Storage (NAS) devices. These essentially are storage devices with a network front end that utilizes the SMB protocol to export the file system to the network. A shared storage solution can provide backup and availability capabilities and is typically always on, whereas a computer or laptop may not always be turned on or may not always be local to the network. A home NAS drive can provide significantly more storage than cloud-based services and can be more economical.

Other communication devices include routers, switches, network gateways, and internet and cellular modems. Typically, these devices do not require network file system support, but many provide storage capabilities as a value add. Communication devices are natural network service hubs. Not only do they provide the gateway to the internet, but they also typically provide essential services for home and office networks. Address assignment, network firewall, and address translation are just some of the services typically provided by these devices. Today, nearly all home and office routers provide USB ports that allow external storage devices to be plugged in.

## Healthcare Use Cases

Healthcare is a very technology-driven market. HIPAA compliance as well as productivity enhancements and patient service are motivating factors in this technology. Healthcare solutions utilize both client and server storage capabilities.

Many NAS vendors specifically target the healthcare marketplace due to the requirements of data security, capacity, performance, interoperability, and compliance. Hospitals and other healthcare networks must deliver high bandwidth access to storage. Physical security of the data as well as this performance requirement makes cloud-based storage problematic.

In addition to the back-room storage requirements, there is a wealth of devices that collect data and need to store this data for later retrieval. The most natural devices in this segment are those involved with Diagnostics and Imaging. MRI, Radiography, Fluoroscopy, Ultrasounds, and EKGs all generate images that must be stored. There are SMB solutions for nearly all these devices.

## Industrial Controls and Construction Use Cases

Many manufacturing business operations rely on internal file sharing with private data servers maintained by their organizations. Not only do their operations depend on data sharing, but many machines are also network aware and use SMB to access dies and models for operation.

As an example, the Phrozen 3D printer uses SMB to retrieve models to print. Injection Molding, Die Casting Machines, and System Robotics all have options to obtain molds and dies over SMB.

Robotics is increasingly being used in construction for repetitive or precision work. Tasks are programmed into these robots by the transfer of files. SMB can be a flexible method to download robotic programs into devices.

## Other Markets

There are many other markets that benefit from the use of file sharing and the SMB protocol. Essentially any use-case that manages images or runs scripts that control devices benefit from sharing files. Defense and avionics, Automotive, and Internet of Things (IoT) are just some. For example, the Reolink PoE (power over ethernet) camera and the Reolink Wi-Fi cameras both support the storage of videos on a local NAS drive using SMB.

## Summary

Hopefully, after reading this article you have a good understanding of what an embedded SMB stack is, and how it can add value to your products. Be sure to read the follow on article, “Considerations for choosing an SMB stack. An SMB feature is typically an add-on to the underlying functions of your product, but it can be a useful and appreciated feature.

If you have questions or comments about this blog or our SMB product, feel free to contact me.

## About the Author

Richard Schmitt, CEO of Connected Way, LLC. and Developer of ConnectedNAS.
rschmitt@connectedway.com
https://www.connectedway.com
Austin, TX

Richard has over 40 years of experience developing software for real-time embedded systems in various roles as CEO, CTO, Engineering Management, and Software Development. He founded Connected Way, formerly Blue Peach in 2002 to provide a multi-platform event-driven SMB stack for embedded platforms. Richard is keenly interested in embedded Linux security, and networking. If he's not answering his email, he's likely at the beach or mountain biking.