Adaptation of a Digital Signage Solution for repurposed TVBox Devices

Luiz F. B. de Araujo, Marcio Oyamada, Guilherme Galante
Center for Exact Sciences and Technology
State University of Western Paraná (Unioeste) – Cascavel Campus
Cascavel-PR, Brazil
{luiz.araujo2, marcio.oyamada, guilherme.galante}@unioeste.br

***Abstract*—The dissemination of information via digital displays has become common in public places and in enclosed spaces with large numbers of people. With the increasing number of displays, local management of content becomes impractical, so flexible digital signage solutions with remote management and availability of content on a server are required. This paper presents the customization of an open source Xibo digital signage system. The focus is on adapting the client for 32- and 64-bit ARM architectures so that it can run on reused TV boxes. This solution is not only environmentally friendly, but also provides a cost-effective solution for deploying digital signage solutions by reusing devices that would otherwise be destroyed.**

***Index Terms*—Decharacterization, hardware reuse, IPTV**

I. Introduction

The dissemination of information plays a crucial role in contemporary society, encompassing a wide range of content such as weather forecasts, news, service schedules, and advertisements. Digital signage has emerged as a viable solution for this purpose, enabling dynamic information delivery through display devices strategically installed in public and private environments [1]. To manage and distribute content in such systems, various software platforms—both proprietary and open-source—are available.

In addition to the software layer, implementing a digital signage system requires a hardware infrastructure that typically includes servers, network equipment, and digital displays. These systems often rely on general-purpose computers, such as x86-based machines or single-board computers (SBCs), to render multimedia content. Given the diverse nature of the displayed content—ranging from web pages and schedules to images and videos—the hardware must meet minimum performance and connectivity requirements to ensure smooth operation.

In Brazil, the Federal Revenue Service frequently seizes TVBox devices sold illegally. While legal TVBoxes provide access to licensed streaming platforms such as Netflix and Apple TV, illicit IPTV devices offer unauthorized access to paid content at no cost or at a reduced price. This practice constitutes piracy and is considered a criminal offense under Brazilian law, specifically Article 184 of the Penal Code [2], [3]. Moreover, the National Telecommunications Agency (Anatel) has raised cybersecurity concerns regarding such devices, noting that many contain vulnerabilities that may compromise personal data and facilitate large-scale cyberattacks [4].

To mitigate these impacts, the Federal Revenue Service has begun donating such devices—after legal clearance—to universities, encouraging their repurposing as low-cost general-purpose computers. This involves removing the original firmware (often a modified version of Android that includes piracy-related applications) and installing a Linux-based operating system tailored for general use.

Within this context, this work proposes the reuse of repurposed TVBox devices in a digital signage system built upon the open-source Xibo platform [5]. The paper outlines the challenges involved in adapting the client application to support 32- and 64-bit ARM architectures and evaluates the feasibility of the proposed solution. The results demonstrate a socially and environmentally beneficial alternative that reduces the need to acquire new equipment while promoting the reuse of hardware that would otherwise be discarded.

II. Digital Signage

Digital signage employs digital displays to present dynamic content in public environments. It distinguishes itself from traditional signage through enhanced interactivity and visual appeal, improving customer experience and potentially increasing sales [6] [7].

A. Digital Signage Solutions

To identify suitable digital signage solutions for this project, searches were conducted on Google and GitHub using the keyword "*digital signage*", focusing on free and open-source alternatives that met the project's requirements. Solutions that were commercial, had fewer than 250 *commits*, or lacked published *releases* were excluded from the analysis. This process revealed three promising options, presented in Table I.

Anthias is a free and open source digital signage system developed specifically for the Raspberry Pi. It is developed in Python and distributed under both the GPLv2 and a commercial license [8]. Although it is a functional option, its server is limited to supporting a single client per instance.

piSignage, in contrast, is a hybrid solution: its server is open-source under the MIT license [9], while the client-side application is proprietary. The system allows up to two clients

Table I
COMPARISON OF CATALOGED SOLUTIONS: ANTHIAS, PISIGNAGE, AND XIBO

Software	Language	License
Anthias	Python	GPLv2 + Commercial license
piSignage	JavaScript, C	MIT (server) + Paid license (client)
Xibo	PHP, C++	AGPLv3 (server and clients for Linux and Windows)

to be connected free of charge, provided they are registered on the developer's website and linked to the server. Additional clients require a paid license of $25.00 per device.

Xibo provides both server and client applications for Linux and Windows under the AGPLv3 license [5], ensuring full access to the source code and no limitations on features or the number of clients. However, versions for Android, Tizen, and webOS are proprietary and require paid licenses. Xibo stands out from the other solutions by offering broader compatibility with general-purpose Linux systems, rather than being tied to a specific SBC (Single-Board Computer), such as the Raspberry Pi, as is the case with Anthias and piSignage. An additional advantage lies in the fact that both its server and clients (for Linux and Windows) are open-source. Consequently, Xibo was selected as the preferred solution for this project.

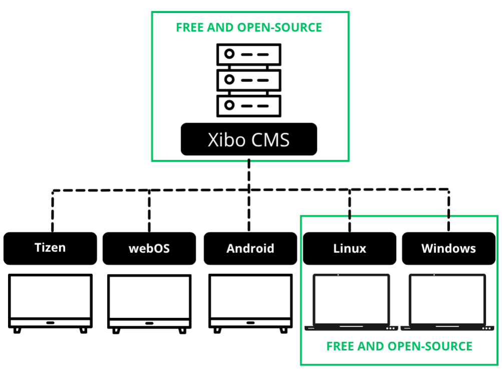

Fig. 1. Xibo Architecture

In Xibo, content can be created graphically through an integrated layout editor, supporting RSS feeds, HTML pages, clocks, text, images, videos, and other media elements. Its clients offer offline playback capabilities in the event of a server disconnection, similarly to piSignage, although they do not support content playback directly from USB devices. The server component, known as Xibo CMS (Content Management System), imposes no restrictions on features or the number of registered clients in its open-source version [5].

B. Related Works

The typical application of repurposed TVBoxes has been in educational contexts. The "Além do Horizonte" (Beyond the Horizon) project, developed by IFSul Minas Gerais, distributed more than 64 repurposed TVBoxes to educational institutions for instructional use [10]. In the same domain, Sobrinho et al. (2024) propose using repurposed TVBoxes as *thin clients* in educational environments.

Another potential application is their use as IoT nodes. Sato et al. (2024) present a people-counting system leveraging TVBoxes running lightweight image recognition algorithms for edge computing.

In contrast to prior work, the present study focuses on adapting seized and repurposed TVBoxes specifically for digital signage applications. Compared to commercial alternatives, the proposed solution—based on an open-source Linux client—offers a low-cost and flexible approach for deploying and maintaining digital signage systems.

III. SELECTED HARDWARE PLATFORMS

In this study, three models were selected for this study (Table II) based on their compatibility with internal Armbian installation and their ability to support a graphical interface. In this work, the In X Plus, TG3, and BTV B11 TVBoxes are referred to as TV1, TV2, and TV3, respectively. TV1 features a relatively outdated SoC based on a 32-bit ARM architecture, while TV2 and TV3 utilize 64-bit processors commonly found in mid-range embedded systems. For reference, the Cortex-A53 used in TV2 is the same processor used in the Raspberry Pi 3B.

Table II
TVBOXES SELECTED FOR THE STUDY

	TV1	TV2	TV3
SoC	Rockchip RK3228A	Amlogic S905X	Amlogic S905X3
CPU	4× Cortex-A7 1.2 GHz	4× Cortex-A53 1.5 GHz	4× Cortex-A55 1.9 GHz
GPU	Mali-400	Mali-450	Mali-G31
RAM	2 GB	2 GB	2 GB
Storage	8 GB eMMC	16 GB eMMC	16 GB eMMC

A. Graphics Software Stack on Linux ARM

A critical component of any digital signage solution is the graphical interface, as all displayed content depends on reliable rendering. This process involves a layered software stack, ranging from low-level video drivers to user-facing toolkits that handle interactive elements such as buttons and menus.

Graphics servers and compositors, such as X11 and Wayland, rely on GPUs and their drivers to render the output. On ARM-based systems, the Lima driver is used for GPUs based on the Utgard architecture, such as the Mali-400 and Mali-450 found in TV1 and TV2. Lima is a kernel-space driver developed through open-source reverse engineering, enabling Linux to interface directly with compatible GPUs. Due to hardware constraints, it only supports OpenGL ES 2.0 and OpenGL 2.1. Its primary advantage is full integration with the Linux kernel, allowing lightweight graphical applications to run without additional configuration [13].

For newer GPUs, such as the Mali-G31 found in TV3, the Panfrost driver is used. Panfrost supports the Midgard and Bifrost architectures and provides broader compatibility with modern graphics APIs, including OpenGL 3.1, OpenGL ES 3.1, and Vulkan 1.1, depending on the specific model. Full OpenGL conformance, however, is only achieved with a subset of these GPUs, such as the Mali-G52, G57, and G610 [14].

In addition to GPUs, some TVBox platforms include dedicated hardware video decoders, such as the Hantro VPU and Amlogic AVE [15]–[17]. To enable smooth video playback requires the cooperation of multiple layers, including hardware, kernel drivers, user-space interfaces, multimedia transcoders, and the application layer.

To complete the software stack, lightweight and customizable graphical environments such as Sway can be used. Sway is a tiling window manager and Wayland compositor that arranges application windows in a logical grid layout, optimizing screen usage and enabling keyboard-based interface control [18].

After the installation of Armbian, GPU performance was assessed using the GLMark2 benchmarking tool. Table III presents the results. For TV1, which integrates the RK3228 SoC, the benchmark scored 41 points. Although this result is low, the rendered output was visually correct, without mesh or color artifacts. GLMark2 under X11 failed to initialize on this device, returning the error: "*main: Could not initialize canvas*."

On TV2, tests were conducted under both X11 and Wayland, with the Wayland session achieving a score 27% higher. On TV3, performance under X11 (XFCE) was inferior to that under XWayland (Sway), where it slightly outperformed Wayland—a result that contrasts with TV2.

Table III
GLMARK2 BENCHMARK RESULTS

Graphical Environment	TV1	TV2	TV3
X11/XFCE	–	–	214
XWayland/Sway	–	313	579
Wayland/Sway	41	399	548

In the video decoding tests, the open-source movie *Big Buck Bunny* (1080p, H.264) via the Xibo Linux client, only TV2 was able to deliver smooth playback with hardware acceleration. TV1 failed to activate the decoder within Xibo, relying solely on CPU decoding and exhibiting low performance. TV3 also showed poor playback quality, attributed to OpenGL incompatibilities.

B. Adapting the Xibo Client for ARM-based Execution

Several adaptations were necessary to enable the Xibo Linux client to operate not only on its original x86 architecture but also on ARM-based platforms, the focus of this work. Initially, the `snap/snapcraft.yaml` file was modified to allow compilation on all architectures supported by Snapcraft, including `arm64`, `armhf`, `powerpc`, `ppc64el`, `s390x`, and `riscv64`. Additionally, the hardware key generation function in `HardwareKeyGenerator.cpp` was changed to rely solely on the device's MAC address, to overcome the incompatibility with the x86-specific CPUID instruction.

Due to Snap Core 20's lack of cross-compilation support, the compilation was performed on a Raspberry Pi 3 running Ubuntu 20.04 for the `armhf` (ARMv7) and `aarch64` (ARMv8) targets.

The Crypto++ and Boost libraries, widely used in C++ applications, were also adapted. Crypto++ provides cryptographic primitives, while Boost extends the standard library with modules for data structures, multithreading, and networking [19], [20]. The Boost version was updated from 1.70 to 1.71 to match the version in the Ubuntu Focal repositories. Similarly, Crypto++ was switched to the native package version, removing the need for manual compilation.

To ensure compatibility, PEM-based key encoding in `CryptoUtils.cpp` was replaced with DER, as the native Crypto++ package lacks PEM support. Base64 handling was also revised to accommodate API changes in Boost. These changes reduced build complexity and compilation time.

A major issue was encountered in layout rendering. Despite successful launch, the Xibo player only rendered the background and logo, while the following sandbox error was logged:

```
GDBus.Error: org.freedesktop.portal.
Error.NotAllowed: This call is not
available inside the sandbox
```

This was traced to missing Snap permissions: the `desktop` and `network-status` plugs [21]. Their inclusion resolved the error but did not restore layout rendering.

Further investigation revealed that version 1.8 R7 of Xibo Linux used Ubuntu Bionic repositories—targeted at Core 18—instead of the expected Core 20 packages. This hindered support for ARM builds, as the Bionic archive is limited to `amd64`. The solution involved migrating to the native WebKitGTK available in Ubuntu Core 20 by updating the dependency in `snapcraft.yaml` and corresponding source files.

Specifically, references to `libwebkitgtk-3.0` were replaced with `libwebkit2gtk-4.0`, and the build scripts (`CMakeLists.txt`) and rendering logic (`WebViewGtk.cpp`) were updated accordingly. Notably, the deprecated function `webkit_web_view_set_transparent` was replaced by `webkit_web_view_set_background_color`, enabling proper rendering.

Following this update, full layout rendering was successfully restored on devices with compatible OpenGL drivers.

IV. RESULTS

The system was tested using the TV2 model, which exhibited the best performance in video playback scenarios. It operated continuously for two months, successfully displaying scheduled content from the Xibo CMS, including weather forecasts, thesis defense calendars, and other institutional information.

All adaptations and source-code modifications required to run Xibo on non-x86 hardware were documented and submitted as pull requests to the official project repository for potential integration into future releases of the platform.

Figure 2 illustrates the deployment of the system in the university dining hall. The content displayed is managed by the institution's press office and used to disseminate information such as weather updates, event schedules, and awareness campaigns.

Fig. 2. Deployment of the system at the university dining hall

V. Conclusions and Future Work

The tests conducted throughout this study—employing customized layouts—demonstrated that the implemented solution adequately fulfills its purpose of displaying multimedia content and institutional information such as weather forecasts and scheduling data. The proposed reuse of devices seized by the Brazilian Federal Revenue Service for digital signage applications proved to be viable from functional, economic, and environmental perspectives. In addition to reducing costs associated with the acquisition of new equipment, the initiative contributes to mitigating the environmental impact caused by electronic waste, by repurposing discarded or underutilized devices.

During the validation phase, the system was tested over a continuous two-month period, reliably executing digital signage campaigns. Furthermore, the solution was showcased at the institution's career fair, effectively demonstrating to the broader community the potential benefits of digital signage technologies and hardware reuse.

Nonetheless, technical challenges were identified during the implementation and validation processes, which should be addressed in future iterations of the project. Suggested enhancements include enabling hardware-accelerated video decoding on Rockchip-based devices (such as TV1), improving compatibility with Amlogic hardware platforms, and extending support to additional TVBox models.

Another promising direction for future work involves expanding the system's applicability beyond digital signage. With proper optimization, these devices could be employed in various low-cost computing scenarios, including interactive terminals, educational platforms, and remote monitoring systems.

The Git repository containing the modified version of the Xibo Linux player[1], along with scripts to automate its installation using the Sway environment[2], has been made publicly available on GitHub. This ensures the transparency and reproducibility of the work, enabling further study, adaptation, or deployment by interested parties.

References

[1] R. Want and B. N. Schilit, "Interactive digital signage," *Computer*, vol. 45, no. 5, pp. 21–24, 2012.

[2] P. A. F. Neto and M. S. Silva, "Direitos autorais e internet: o streaming ilegal de obras audiovisuais," *Cadernos de Prospecção*, vol. 12, no. 5, pp. 1190–1190, 2019.

[3] Brasil. (2003) Lei nº 10.695, de 1º de julho de 2003. [Online]. Available: https://www.planalto.gov.br/ccivil3/LEIS/2003/L10.695.htm

[4] R. Péret. (2023) Novas punições para tv pirata com multa até para os usuários. [Online]. Available: https://tribunaonline.com.br/economia/novas-punicoes-para-tv-pirata-com-multa-ate-para-os-usuarios-145459

[5] Xibo. (2024) Open source digital signage. Xibo. [Online]. Available: https://xibosignage.com/open-source

[6] C. Bauer, P. Dohmen, and C. Strauss, "Interactive digital signage-an innovative service and its future strategies," in *2011 International Conference on Emerging Intelligent Data and Web Technologies*. IEEE, 2011, pp. 137–142.

[7] C. Dennis, R. Michon, J. J. Brakus, A. Newman, and E. Alamanos, "New insights into the impact of digital signage as a retail atmospheric tool," *Journal of consumer Behaviour*, vol. 11, no. 6, pp. 454–466, 2012.

[8] Screenly. (2022) Anthias. Screenly. [Online]. Available: https://anthias.screenly.io/

[9] piSignage. (2024) pisignage. [Online]. Available: https://pisignage.com/

[10] D. da Cunha Mendes and F. Corsini, "O impacto social do projeto além do horizonte nas escolas públicas municipais beneficiadas," *17º JORNADA CIENTÍFICA E TECNOLÓGICA E 14º SIMPÓSIO DE PÓS-GRADUAÇÃO DO IFSULDEMINAS*, vol. 15, no. 3, 2023.

[11] M. P. Sobrinho, J. V. M. Sousa, C. A. N. da Silva, L. R. Santos *et al.*, "Explorando tv boxes em ambientes educacionais: Vantagens e limitações operando como thin client," in *Escola Regional de Computação do Ceará, Maranhão e Piauí (ERCEMAPI)*. SBC, 2024, pp. 249–258.

[12] G. P. da Luz, G. M. Sato, L. F. G. Gonzalez, and J. F. Borin, "Repurposing of tv boxes for a circular economy in smart cities applications," *Scientific Reports*, vol. 15, no. 1, p. 22638, 2025.

[13] Mesa3D. (2024) Lima. Mesa3D. [Online]. Available: https://docs.mesa3d.org/drivers/lima.html

[14] ——. (2025) Panfrost. [Online]. Available: https://docs.mesa3d.org/drivers/panfrost.html

[15] Rockchip. (2015) Rockchip rk3228a datasheet. Rockchip. [Online]. Available: http://pro934ad4.pic3.ysjianzhan.cn/upload/RockchipRK3228ADatasheetV1.pdf

[16] Amlogic. (2017) S905x datasheet. [Online]. Available: https://datasheet4u.com/datasheet-pdf/AMLOGIC/S905X/pdf.php?id=1410887

[17] ——. (2020) S905x3 datasheet. [Online]. Available: https://datasheet4u.com/datasheet-pdf/Amlogic/S905X3/pdf.php?id=1483651

[18] Sway. (2024) Sway. Sway. [Online]. Available: https://swaywm.org/

[19] B. I. Tuleuov and A. B. Ospanova, "Crypto++," in *Beginning C++ Compilers: An Introductory Guide to Microsoft C/C++ and MinGW Compilers*. Springer, 2024, pp. 149–156.

[20] S. Koranne, "Boost c++ libraries," in *Handbook of open source tools*. Springer, 2010, pp. 127–143.

[21] R. Ancell. (2022) Help: Gdbus.error this call is not available inside the sandbox. [Online]. Available: https://forum.snapcraft.io/t/help-gdbus-error-this-call-is-not-available-inside-the-sandbox/29969/5

[1] https://github.com/lbecher/xibo-linux

[2] https://github.com/lbecher/xibo-client-config