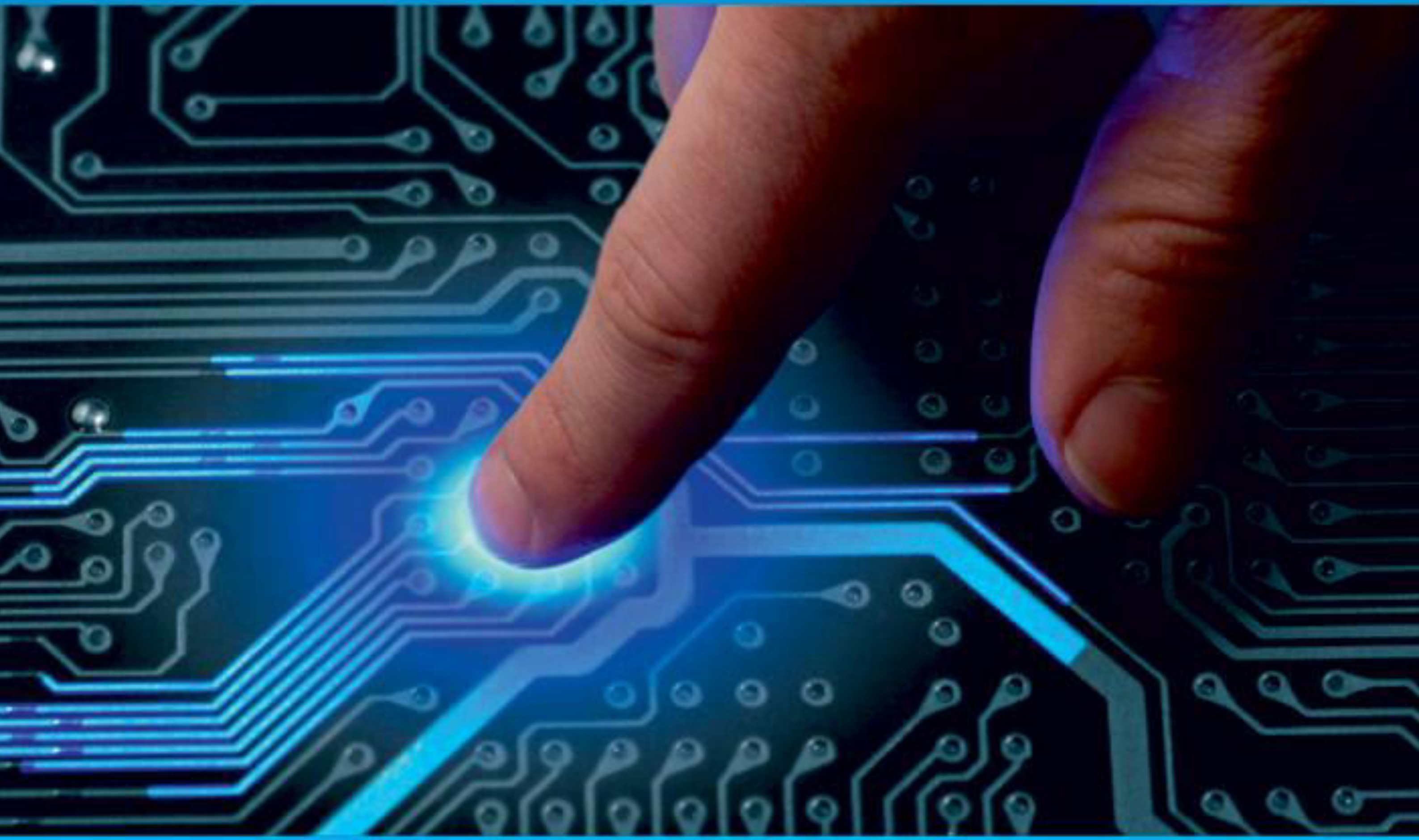
IJIRCCE

e-ISSN: 2320-9801 | p-ISSN: 2320-9798



INTERNATIONAL JOURNAL OF INNOVATIVE RESEARCH

IN COMPUTER & COMMUNICATION ENGINEERING

Volume 11, Issue 7, July 2023

INTERNATIONAL STANDARD SERIAL NUMBER INDIA

Impact Factor: 8.379

International Journal of Innovative Research in Computer and Communication Engineering

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 8.379 |

|| Volume 11, Issue 7, July 2023 ||

| DOI: 10.15680/IJIRCCE.2023.1107069 |

Embedded Linux-based Client-Server Upgrade Model of Design and Realization

B R Bhargava, Usha J

PG Student, Dept. of MCA, R V College of Engineering, Bengaluru, India

Professor, Dept. of MCA, Dept. of MCA, R V College of Engineering, Bengaluru, India

ABSTRACT: The rapid evolution of embedded systems has created a demand for efficient and seamless upgrade models to enhance the functionality and performance of client-server architectures. This paper presents an innovative approach, the Embedded Linux-based Client-Server Upgrade Model, designed to address the challenges of system upgrades in embedded environments.
The proposed model leverages the flexibility and robustness of the Linux operating system, making it an ideal foundation for managing the complexities of client-server interactions. By integrating Linux into the embedded systems, the model provides a scalable and adaptable platform that enables effortless updates and feature enhancements.

KEYWORDS: Embedded System ; Linux; Program Designing; Finger daemon;

I. INTRODUCTION

Today, over 90% of the processor market share belongs to embedded systems, which are used in many facets of daily life and work. Embedded Linux incorporates features of an embedded operating system in addition to inheriting open-source code from online sources. The embedded Linux core is dependable, efficient, and can handle a wide range of applications. One supported the main flow processors and hardware platforms. When upgrading and digging up programs in embedded Linux systems, we must first download the application program to the target machine. The usual one-to-one download model takes a lengthy time when there are several target machines. The client/server download strategy, which is helpful for monitoring and can save a lot of time, is introduced in this work. [1].

II. LINUX FINGER DAEMON

Linux processes can choose between foreground and background execution. While a foreground process that is active communicates with the person using the terminal operates independently in the background. Although the user can monitor its status, they are unaware of what it is doing. The term "daemons" refers to background-operating processes. A server is a program that executes continually, frequently without interruption or restart, waits for requests to arrive, responds to them, and frequently launches further programs to deal with the requests.

Numerous finger daemon services are launched by Linux during the boot process. In Linux, each interface by which system communicate with the user known as the terminal, each process running from the terminal will be attached to it, known as the process control terminal. When the control terminal is closed, the corresponding processes. will automatically shut down. However, the finger daemon was able to break through limitation above, it is executed starting from the operation until the entire system is shut down. Finger daemon is off terminal and running in the background. To prevent the display of information on any terminal and ensure uninterrupted operation of the finger daemon, it is advisable to detach the finger daemon from the terminal. By doing so, the finger daemon remains unaffected by any terminal information and can continue its functioning undisturbed. If you require a process that remains impervious to user interactions, terminal activities, and other changes, it is essential to design it as a finger daemon. In the context of the server control target machine upgrade program, it has been specifically developed as a finger daemon to maintain control and stability during the upgrade process.

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 8.379 |

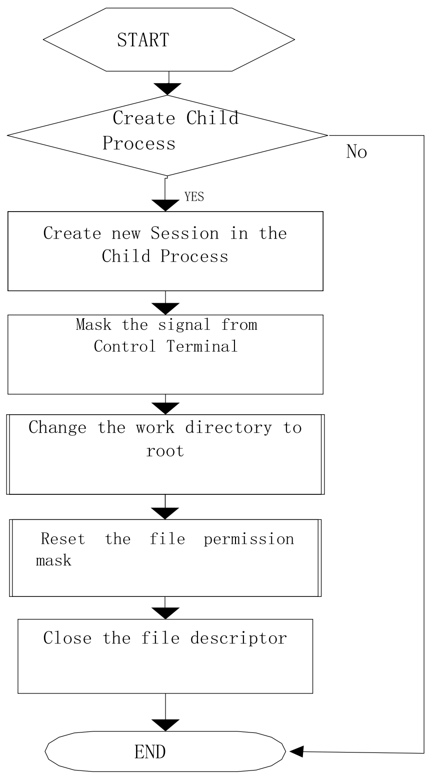
|| Volume 11, Issue 7, July 2023 ||

| DOI: 10.15680/IJIRCCE.2023.1107069 |

III. PROCEDURE

- In order to start the finger daemon in Linux, you can use one of the following methods:
- The upgrade program can be initiated through a guided process by configuring an initialization script. This method is commonly used in Linux to start the upgrade program for various base servers..
- The xinetd finger daemon is capable of launching the upgrade program. Acting as a powerful finger daemon, xinetd is responsible for monitoring all network requests and triggering the corresponding network server program when a request is received.
- The cron utility can also be utilized to start the upgrade program. Cron provides scheduling functionality, allowing server programs to be started at specified times or intervals.
- The start of the upgrade program can be configured within the At finger daemon.
- Manually starting the finger daemon from the console is typically performed in scenarios where the finger daemon has been stopped temporarily or when changes have been made to the service configuration file. By restarting the finger daemon from the console, the configuration file is reloaded, enabling the new configuration settings to take effect. Manual console start-up is often employed for debugging purposes in relation to the finger daemon.

Setup the finger daemon of the basic flowchart is showed in Figure 1.


START
Create Child Process
No
YES
Create new Session in the Child Process
Mask the signal from Control Terminal
Change the work directory to root
Reset the file permission mask
Close the file descriptor
END

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 8.379 |

|| Volume 11, Issue 7, July 2023 ||

| DOI: 10.15680/IJIRCCE.2023.1107069 |

IV. Linux Network Communication

A network is an excellent option for all communication methods. It is more convenient and efficient than serial when the number of target machines is large. The network speed is superior to others.

UDP uses a simple transmission architecture with no associated hand-shaking interactions to assure dependability, ordering, or data integrity. As a result, UDP provides inconsistency, with datagrams arriving out of order, appearing duplicated, or going missing without notice. UDP argues that error checking and correction are either unneeded or built within the application, removing the complexity associated with such processing at the network interface level. Dropping packets is preferable to waiting for delayed packets, which may not be an option in a real-time system. If error correction services are required at the network interface level, an application may employ the Transmission Control Protocol or the Stream Control Transmission Protocol, which are both developed for this purpose.

TCP serves as a bridge between an application program and the Internet Protocol. That instance, instead of splitting up a huge chunk of data and issuing a series of IP queries, an application program can issue a single request to TCP and let TCP handle the IP specifics.

TCP is a dependable stream delivery service that ensures that a data stream transmitted from one host to another is delivered without duplication or data loss.
This module implements communication between the server and target machines using TCP/IP rather than UDP. The server delivers the Upgrade command to the target machines, who receive it and correctly detect it before downloading the upgrade package. Figure 2 depicts the flow of communications.

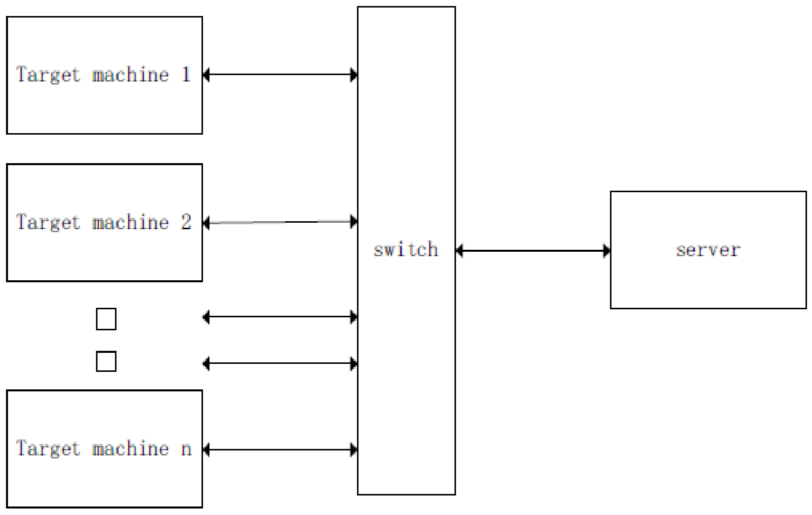


Figure 2 Communication Flow

After the download is complete, unzip the upgrade script to obtain the upgrade script. Figure 3 depicts the program flowchart.
Download the update package from the server; if using shell scripts, this shell script should have executive authority. After downloading the upgrade package, extract the shell script and test the executive permission upgrading script. If it does not have executive authority, the upgrading application should alter the script property by calling system services. After downloading the update package from the server, the target machines should communicate completion information to the server for statistics management.

V. SHELL SCRIPT

This module includes upgrade script and downloader script.

A. Script for Download

This module handles downloading update packages from the server. This study primarily uses the TFTP or FTP protocol for high-speed network transfer. The Trivial File Transfer mechanism is a basic mechanism for transferring files. It is built on top of the User Datagram Protocol and uses port number 69. Because TFTP is

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 8.379 |

|| Volume 11, Issue 7, July 2023 ||

| DOI: 10.15680/IJIRCCE.2023.1107069 |

supposed to be tiny and simple to implement, it lacks the majority of the functionalities of a standard FTP. TFTP only reads and writes files from and to a remote server. It cannot list directories and does not currently support user authentication. The TFTP transport program is simple to implement, and its software costs less than FTP, which is especially essential for embedded devices with limited storage space.

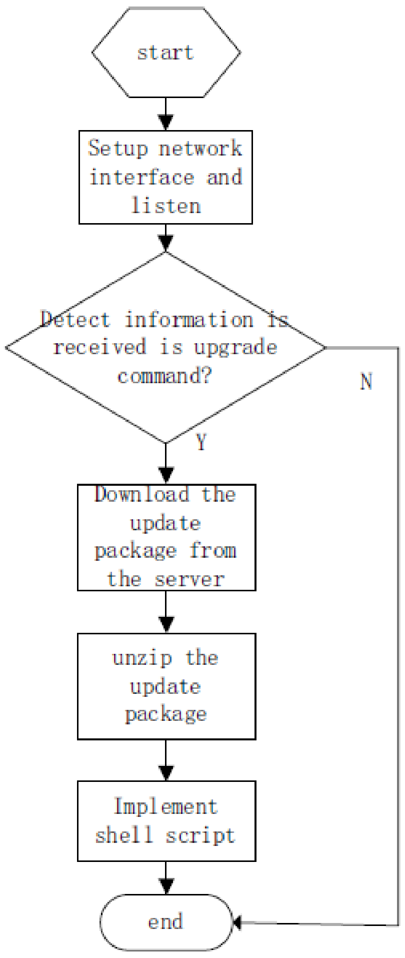

start
Setup network interface and listen
Detect information is received is upgrade command?
N
Y
Download the update package from the server
unzip the update package
Implement shell script
end

Figure 3 Download Flow Chart

This tutorial does not cover how to create a TFTP download program using shell or C. The File Transfer Protocol (FTP) is a network protocol that is used to transfer files from one host to another via a TCP-based network, such as the Internet. FTP is designed as a client/server system, with distinct control and data connections between the client and server. FTP users can connect anonymously if the server is set to allow it, but they must authenticate themselves using a clear-text sign-in protocol. The following is the FTP transport script:

```
>ftp -n<<!
>open server_ip_address>username password >binary
>cd /
>lcd /home/user>prompt
>get >update.tar>close
>bye
!
```

The update.tar is located in the directory specified by the 'cd' command. Using the 'lcd' command, the user can provide the location where the update package should be unzipped.

| e-ISSN: 2320-9801, p-ISSN: 2320-9798| www.ijircce.com | |Impact Factor: 8.379 |

|| Volume 11, Issue 7, July 2023 ||

| DOI: 10.15680/IJIRCCE.2023.1107069 |

B. Script Upgrading

This module is essential for advancement. The module mentioned above is complete. The upgrade process begins by killing necessary processes, then adjusts the system's startup parameters, removes the original software, and copies the new application. You can delete the download upgrade package to free up disk space and restart the target machine. Figure 4 depicts the program flowchart

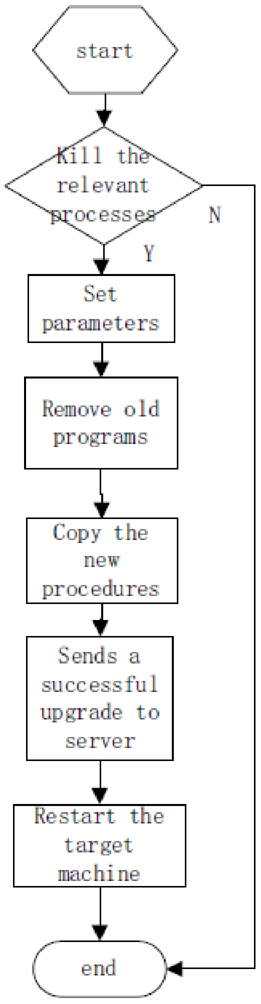

start
Kill the relevant processes
N
Y
Set parameters
Remove old programs
Copy the new procedures
Sends a successful upgrade to server
Restart the target machine
end

Figure 4 Depicts the Program Flowchart

VI. Conclusion and Future Work

Embedded Linux is utilized in embedded systems and can be found everywhere. Typically, the developer must save time during the development process. The user of an embedded system requires a convenient and quick upgrade program. When the number of target machines is very large, the one-to-one model takes a long time, and the c/s model can save a lot of time. This paper describes an upgrading program based on embedded Linux. The concept can also be used to various types of embedded systems. It is highly useful and practical for troubleshooting and updating. This paper does not discuss security can defensive hacker and malicious software, because programmer does not need security protect in development process. But if you consider and design the security of course is very good.

References

[1] A. Robbins, N.H.F. Beebe. Tutorial on Shell Scripting in Linux. 2009.4
[2] Y. Lin, L. Guo. Linux Network Programming. 2001.11
[3] Z. Wang, J. Bao. Prospects of Embedded Linux Systems and Their Applications [J]. Monolithic Integrated Circuit Single-chip Computer and Embedded System Application, 2004(05)

[4] M. Jin, X. Zhou, and L. Jin. Embedded System: Components, Principles, Design and Programming, Posts & Telecom Press, China,2007.
[5] K. Zhao, L. Zhu. Study on Introduction of Linux Curriculum in IT Specialties at Universities. Technology and Market,2009(04)
[6] H. Xu, J. He. Linux Kernel-based Operating System Experiments Guidance, Tsinghua University Press, 2009
[7] C-LAB Cooperative Computing & Communication Laboratory. Real-Time Linux, University of Paderborn, by M. Eikermann, 2001
[8] D. Mao, X. Hu. Analysis of Source Code in the Linux Kernel, Zhejiang University Press, 2003
[9] J. Ni. Linux Kernel Analysis & Programming, Publishing House of Electronics Industry, 2005