Warszawa, 05.06.2024

2024 Linux w Systemach Wbudowanych – Laboratorium ćw. 3

Student: Piotr Jankiewicz

Treść zadania

1. Przygotowanie administracyjnego obrazu systemu Linux (podobnego do używanego podczas laboratorium systemu ratunkowego).
2. Przygotowanie użytkowego systemu Linux pracującego z systemem plików ext4 na drugiej partycji, realizującego jedną z poniższych funkcji
    a. Serwer WWW (zrealizowany na bazie jednego z serwerów HTTP oferowanych w Buildroot, albo jako aplikacja webowa w środowisku Flask, Tornado lub podobnym). Serwer powinien udostępniać pliki z partycji 3 na karcie SD, wyświetlając listę tych plików i pozwalając na wybranie pliku do załadowania. Serwer powinien umożliwiać uwierzytelnionym użytkownikom wgrywanie nowych plików na tę partycję
3. Przygotowanie bootloader'a, umożliwiającego określenie, który system (administracyjny czy użytkowy) ma zostać załadowany.

Procedura odtworzenia projektu z załączonego archiwum

Odtworzenie projektu jest czasochłonne.

1. Zbuduj obraz administracyjny z konfiguracją .config_admin
2. Zbuduj obraz użytkowy z konfiguracją .config_user
3. Wgraj system administratora na partycje pierwsza i stwórz dwie nowe partycje. Jedna sformatuj a na druga wgraj system użytkowy.
4. Dodaj aplikacje i skrypty na system użytkowy.
5. Uruchom ponownie. Serwer funkcjonuje.

Pliki w archiwum:

- .config_admin – plik konfiguracyjny buildroota dla systemu administracyjnego
- .config_user – plik konifuguracyjny buildroot dla systemu użytkwogo
- App.py - aplikacja python3, serwer w technologii flask.
- Index.html - templatka dla flaska

- Boot – plik ze skryptem dla u-boot
- S99web_server – skrypt uruchamiajacy serwer

Opis rozwiązania

Ćwiczenie udało się zrealizować w pełni. Efektem końcowym był serwer, który automatycznie uruchamiał się przy starcie systemu o ile użytkownik nie wciskał ani przycisku pod pinem GPIO 21 lub 25. W takim wypadku serwer uruchamiał się w systemie użytkowym. Serwer umożliwiał podgląd i wgranie plików do systemu plików zmontowanego pod trzecią partycją na karcie SD po podaniu hasła przez interfejs internetowy. Pliki zachowywały się na partycji po ponownym uruchomieniu.

Partycja dla systemu użytkownika i przechowywania plików serwera została przygotowana z poziomu systemu administracyjnego, który został przy konfiguracji wyposażony w niezbędne do tego narzędzia.

System administracyjny został skonfigurowany z wykorzystaniem bootloadera U-boot, który zanim uruchomił system, sprawdzał czy użytkownik wciska przycisk, który indykował, że powinien zostać uruchomiony system administracyjny, czy użytkowy. Należy rozróżnić, że u-boot to co innego niż system ratunkowy dostarczony przez prowadzących przedmiot.

Opis modyfikacji i konfiguracji Buildroota

Konfiguracja BR dla administratora:

1. Początkowo konfigurujemy jak w przewodniku po laboratorium
2. Dodajemy do konfiguracji narzędzia do:
    a. Partycjonowania – fdisk i parted
    b. Do formatowania – e2fsprogs i mkfs
    c. Do kopiowania – rsync i dd
    d. Do naprawiania systemu plików e2fsprogs
3. Dodajemy opcję u-boot zgodnie z wykładem 5. Tworzymy skrypt.

4. Budujemy i wgrywamy system na kartę z poziomy systemu rescue na partycję p1.
5. Uruchamiamy system bez rescue, wiec korzystamy z utworzonego systemu administracyjnego.
6. Początkowo tworzymy dwie nowe partycje p2 i p3. P3 formatujemy do postaci wskazanej w ćwiczeniu.
7. Tak przygotowany system i karta, może teraz nam posłużyc do wgrania systemu użytkowego.
8. Ze stacji roboczej tworzymy plik boot na podstawie skryptu boot dl u-boot zgodnie ze wskazaniami z wykładu 5.

Konfiguracja BR dla użytkownika:

1. Początkowo konfigurujemy zgodnie z przewodnikiem po laboratorium, jednak nie używamy initramfs.
2. Dodatkowo dodajemy pakiet python3 i flask.
3. Budujemy obraz użytkowy.
4. Wgrywamy obraz na partycję drugą.
5. Dodajemy do /etc/inid.d skrypt uruchamiający serwer flask przy pomocy python3. Umieszczamy aplikacje w systemie przy pomocy wget. Dodajemy folder “templates” i umieszczamy w nim index.html. Jest to folder wymagany przez flask. Trzeba pamiętać o zmianie chmod+x na pliku app.py.

Tak skonfigurowany system bez ingerencji przy uruchamianiu samodzielnie startuje serwer, który umożliwia zautoryzowane przechowywanie plików na trzeciej partycji.