OpenVPN szerver kezelő

A szerver kezelő implementálása bottom-up programozási stílusban készült. Több modulra lett bontva a folyamat, amely a végén lett az elkészült modulok segítségével egyesítve.

Az implementálás kezdetén létrejött a linux nevezetű modul kezdeti verziója, amely alapvető Linuxszal kapcsolatos problémákat old meg, és az implementálás során bővítésre került az alkalmazási igénynek megfelelően.

A linux modul több utility funkciót is tartalmaz:

Legelőször is az exists funkció került implementálásra, amely ellenőrzi, hogy egy fájl, vagy egy jegyzék létezik-e.

Egy paramétert vár, amely a jegyzék vagy a fájl elérési útvonala. Fontos, hogy jegyzék esetén az elérési útvonal vége / vagy \ jel legyen.

A feladat ellátására a Lua beépített OS függvénykönyvtárának a rename funkciót használja.

A rename beépített funkciót úgy használja, hogy a fájlt/jegyzéket saját magára próbálja elnevezni. Három visszatérési értékkel is rendelkezik a rename funkció.

Az első visszatérési érték maga a művelet sikeressége (igaz/hamis érték). A második visszatérési érték a művelet sikertelenségekor a hibaüzenetet tartalmazza általában angolul, a harmadik pedig a hiba kódját. Több lehetőség is fennállhat, ebből kettő eset fontos nekünk:

* ha nem létezik a fájl/jegyzék, akkor szimplán az első visszatérési értéke nem lesz true vagyis igaz, és a hibakód is más lesz (például hibakód 2; file and directory doesn’t exist)
* ha létezik a fájl/jegyzék, és saját magára elnevezzük, akkor a művelet sikeres lehet, ezáltal az első visszatérési érték true lesz
* ha létezik a fájl/jegyzék, akkor sem garantált mindig az átnevezés sikeressége, ezért kaphatunk például hamis első visszatérési értéket az átnevezési művelet sikertelenségéről, és egy hibakódot. Ilyen hibakód lehet például a 13-as hibakód, amely a Permission Denied vagyis hozzáférés megtagadva. 17-es hibakóddal is visszatérhet a rename, ekkor hálózati forrással kapcsolatos hibára jut.

Összességében, ha az os.rename funkció igazzal tér vissza akkor létezik a fájl/jegyzék, vagy ha 13-as és 17-es hibakóddal tér vissza, akkor is létezik a fájl/jegyzék.

A következő legfontosabb alappillér az exec\_command\_with\_proc\_ret\_code funkció.

Ez a funkció három paramétert fogad:

* a legelső paraméter maga a Shell-parancs, amit le szeretnénk futtatni
* a második paraméter megadja, hogy szeretnénk-e, ha visszaadná a funkció a futtatott shell parancs kimenetének szövegét
* a harmadik paraméter pedig a környezeti változók objektuma

A környezeti változók a parancs legelejére kerülnek beillesztésre, az export shell parancs segítségével. Az export shell parancs (KÖRNYEZETI\_VÁLTOZÓ\_NEVE=érték) formátumban fogadja a paramétereket.

A parancs után pedig beillesztésre kerül egy echo $? shell parancs, mert a régebbi Lua verziókban csak így lehet megkapni egy processz/parancs visszatérési értékét.

A Lua io (input-output) függvénykönyvtárat használja fel, és abban is a popen (processz open) funkciót. Ez a függvény 2 paramétert fogad el, az első maga a futtatandó parancs, a másik pedig, hogy olvasni/írni akarunk-e a processznak. Futtatás után egy file-handlet kapunk, amivel tudjuk kezelni az adott processz kimenetét.

A kapott file-handlera flush funkciót futtat le annak érdekében, hogy biztosan megérkezzen az összes adat a programhoz. Ezután a lines funkcióval beolvassuk a sorokat, amelyeket a futtatott parancstól kaptunk. Fontos, hogy a kapott file handlet a beolvasás után bezárjuk, megsemmisítjük. Az utolsó sor tartalmazza a visszatérési értéket, ezt külön eltároljuk. A kapott output sorokból kitöröljük az utolsó sort.

Visszatérési értéke függ a második paramétertől, ha nincs beállítva, akkor csak a parancs exit codeját adja vissza, ha igazra van állítva, akkor a legelső visszatért érték a parancsfuttatás eredménye, a második pedig a parancsfuttatás exit codeja.