Jun 2017. **Broj 47**

LiBRE!

Časopis o slobodnom softveru

JOŠ IZDVAJAMO

Način povratka izgubljenih podataka

Jezička laboratorija u Raketu

Creative Commons Autorstvo-Nekomercionalno-Deliti pod istim uslovima

Reč urednika

Odgovor je važan

Prošlo je tri meseca od objavljivanja prošlog broja. Iz našeg ugla ova razlika između dva broja je pomalo velika. Iako nam nije bila namera da toliko vremena protekne do novog broja, ipak smo na kraju, kad je trebalo da se presudi da li je bitniji kvalitet ili trenutak objavljivanja, ostali verni zacrtanom cilju. Zato ćete u ovom broju naići na veći broj raznovrsnih tekstova i poprilično smo sigurni da će čitaoci svih profila, i početnici i profesionalci, naći bar jedan interesantan članak, a verovatno i više od jednog.

Kao glavni cilj ostaje nam zacrtan podsticaj i promocija slobodnog softvera kod nas i u regionu. Pošto je i sam Časopis projekat zajednice, potpuno smo svesni važnosti odgovora koji jedan projekat očekuje od zajednice. Zato želimo da podsetimo naše čitaoce da se sudbina ovog projekta kao i svih drugih projekata slobodnog softvera nalazi u njihovim rukama. Moć korisnika slobodnog softvera se ne nalazi samo u mogućnosti da on svojim tehničkim znanjem učestvuje u projektu, već se njegova moć može izraziti kroz jednostavan odgovor autoru projekta. Odgovor je važan jer on ima mogućnost da promeni tok projekta. Nemojte da vas vaš nedostatak tehničkog znanja zaustavi u tome da predložite nekakve promene timu koji radi na razvoju određenog softvera ili na održavanju projekta kao što je LiBRE! časopis.

Na samom početku projekta odgovor zajednice je najbitniji i može najviše da utiče na dalji razvoj, a u ovom broju imamo čast da vam predstavimo nedavno objavljenu linuks distribuciju Ekvilibrijum. Ova distribucija računa na odgovor zajednice. Mi kao časopis smo se potrudili da je što pre prikažemo čitaocima, a na vama ostaje da učestvujete u stvaranju puta kojim će se razvijati. Pored ovog projekta opet ćemo se osvrnuti, uz malo detaljniji opis nego ranije, na jednu od najpoznatijih konferencija u regionu Dors/Cluc, kojoj je ovo impresivno dvadeset četvrto održavanje, a ove godine je član redakcije prisustvovao jednoj od najpoznatijih evropskih konferencija FOSDEM, tako da vas očekuje izveštaj sa nje. Za početnike u ovom broju smo spremili nastavak serijala o Gnu-Linuks naredbama, uvod u Blender, a za one zainteresovane za programiranje započinjemo serijal Uvod u objektno orijentisano programiranje u C++. Međutim, ovaj serijal nije jedini koji će biti interesantan programerima. Jezička laboratorija

u Raketu predstavlja još jedan novi serijal koji će čitaoce provesti kroz proces stvaranja programskog jezika. Imaćete priliku da pročitate nastavak sage o kripto-ratovima, kako povratiti izgubljene podatke, i još mnoge druge zanimljive teme.

Za ovaj broj smo uspeli da nađemo lokalne projekte za predstavljanje, a pripreme za sledeći broj su već u toku. Mi smo u konstantnoj potrazi za projektima koje vredi predstaviti lokalnoj zajednici, a od vas bismo voleli saznati o kojim lokalnim projektima biste voleli da čitate u Časopisu, a mi ćemo se potruditi da svim projektima pružimo punu pažnju. Pišite nam na već poznatu adresu libre@lugons.org.

Do sledećeg broja,

LiBRE! tim

Sadržaj

Vesti str. 6

Puls slobode
Fosdem 2017 str. 10

Predstavljamo
Dors/Cluc Konferencija str. 17
Ekvilibrijum str. 22

Kako da...?
Način povratka izgubljenih podataka - Fotorek str. 27
Prvi koraci u Blenderu str. 35
Uvod u objektno orijentisano programiranje C++ str. 39
Ki-Pas iz komandne linije str. 43
Numerička obrada i simulacije str. 49

Oslobađanje
Naredbe u GNU-Linuksu str. 51

Slobodni profesionalac
Fan - Nagios str. 55
Jezička obrada u Raketu str. 59

Internet, mreže i komunikacije
Kripto-ratovi: Nekada i sada str. 65

Mobilni kutak
Šifrovano ćaskanje na Androidu: Ksaber str. 69

Moć slobodnog sofvera

Libre! prijatelji

Broj: 47

Izvršni urednik: Nikola Todorović

Glavni lektor:
Admir Halilkanović

Lektura:
Jelena Munćan, Saška Spišjak, Aleksandar Božinović

Grafička obrada:
Dejan Maglov, Zoran Lazarević

Dizajn: White Circle Creative Team

Autori u ovom broju:
Nikola Todorović, Zlatko Barnak

Marjan Đuran, Amar Tufo, Adrijan Đurin, Ivana Isadora Devčić, Luka Hadži-Đokić, Milan Stojakov, Stefan Biševac, Petar Simović, Stefan Nožinić

Počasni članovi redakcije:
Željko Popivoda, Mihajlo Bogdanović, Vladimir Popadić, Željko Šarić, Aleksandar Stanisavljević

Kontakt:
IRC: #floss-magazin na irc.freenode.net

E-pošta: libre@lugons.org
Veb: http:/libre.lugons.org

Vesti

3. mart 2017.

NASA objavila kod svog softvera

Nacionalna vazduhoplovna i svemirska agencija (NASA) je objavila katalog softvera, koji pored softvera isključivo namenjenog za potrebe ove agencije, sadrži i veliki broj alata i biblioteka koje mogu da budu od pomoći preduzetnicima, akademskim ustanovama, malim preduzećima i industriji.

Korisni link: http://bit.ly/2qvJNEY

16. mart 2017.

SUSE ekspert dan u Beogradu

Ovog marta u Beogradu je prvi put održan SUSE dan eksperata na kom su predstavljeni novi alati za sistem-administratore i demonstrirana njihova upotreba.

Korisni link: http://bit.ly/2rP8m3p

18. mart 2017.

Dan instalacije linuksa

U subotu 18. marta od 10 do 14 časova na Fakultetu tehničkih nauka (FTN) održao se Dan instalacije linuksa (eng. *Linux Install Day*). Događaj je prvenstveno bio namenjen studentima, ali je bio otvoren za sve koji su želeli da prisustvuju.

Korisni link: http://bit.ly/2qvE2XN

25. i 26. mart 2017.

Tilda FriBSD događaj

Na Departmanu za matematiku i informatiku Prirodno-matematičkog fakulteta u Novom Sadu, u organizaciji Fakulteta i Tilda hakerspejsa, održan je niz predavanja koja su za temu imala FriBSD operativni sistem i alate srodne njemu. Sledećeg dana su se održala radionica gde su učesnici mogli da isprobaju tehnologije spominjane na predavanjima.

Korisni link: http://bit.ly/2qi7IHE

29. mart 2017.

Dan slobode dokumenata

Dvadeset i devetog marta je obeležen dan slobode dokumenata koji predstavlja godišnji događaj koji „slavi otvorene standarde i podiže svest o njihovom postojanju”.

Korisni link: http://bit.ly/2rdKjbB

4. april 2017.

Mastodon

Pojavila se nova decentralizovana društvena mreža otvorenog koda pod nazivom Mastodon. Ova mreža je jako brzo stekla popularnost kao decentralizovana alternativa Tviteru.

Korisni link: http://bit.ly/2nVXYAT

Vesti

5. april 2017.

Najava promene na Ubuntuu

Nakon sedam godina korišćenja Junitija kao osnovnog desktop okruženja, jedna od najpopularnijih linuks distribucija Ubuntu će se od 18.04 verzije vratiti na Gnom desktop okruženje. Ova promena se očekuje u toku sledeće godine, a pored ove promene Kanonikal je odlučio da prestane sa razvojem svog Ubuntu telefona.

Korisni link: http://bit.ly/2s7QhuP

13. april 2017.

Ubuntu 17.04

Objavljeno je novo stabilno izdanje Ubuntu operativnog sistema. Zajedno sa 4.10 verzijom kernela, u 17.04 verziji Ubuntua uključena je i podrška za štampače bez drajvera (eng. *driverless printers*). Najznačajnije promene su vezane za klaud tehnologiju i servere.

Korisni link: http://bit.ly/2qzWAp8

18. april 2017.

Kafe 2

Fejsbuk je objavio novi frejmvork za duboko učenje (eng. *deep learning framework*) pod nazivom Kafe2. Ovaj frejmvork je objavljen pod BSD licencom i dostupan je na Githabu.

Korisni link: http://bit.ly/2rykGW9

10. maj 2017.

Tanderberd

Za 52. izdanje Tanderberd mejl klijenta, pored vizuelnih promena interfejsa, značajna je i mogućnost uklanjanja mejl naloga uz brisanje svih podataka vezanih za taj nalog. Mozila Fondacija će nastaviti da pruža organizacionu podršku ovom projektu.

Korisni link: http://bit.ly/2qzVoSD
http://bit.ly/2rgY77B

25. maj 2017.

Devuan 1.0

Dostupno je prvo stabilno izdanje Devuan distribucije. Ova distribucija je bazirana na Debijanu i nastala je iz potrebe za uklanjanjem sistem-D-a.

Korisni link: http://bit.ly/2ryi6PN

Puls slobode

FOSDEM'17

Izveštaj sa konferencije

Autor: Marjan Đuran

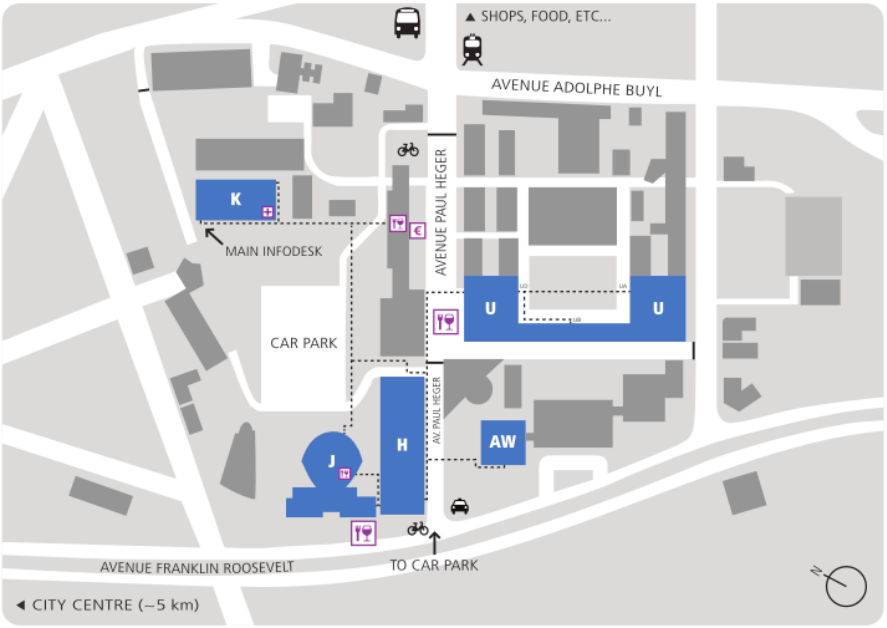
O FOSDEM-u

FOSDEM je najveća konferencija koja promoviše besplatan softver otvorenog koda u Evropi. Održava se u Briselu (Belgija) i, kako kažu, cela organizacija ove konferencije zasniva se na volonterskom radu i želji svesnih ljudi da rade na rasprostranjivanju softvera otvorenog koda. Kada kažemo „konferencija" uglavnom asocijacija bude na predavače u odelima i lepo uređene svečane sale, međutim FOSDEM nije ni nalik tako nekom „uštogljenom" događaju, već se naprotiv i reklamira kao nekomercijalna konferencija. Ideja FOSDEM konferencije je da se programeri iz cele Evrope i sveta tu upoznaju, a možda i započnu saradnju ili projekat, kao i da se predstave novi zanimljivi projekti iz sveta besplatnog softvera otvorenog koda i održi korak sa najnovijim zbivanjima i trendovima. Konferencija je prvi put organizovana 2000. godine kada je Raphael Bauduin, jedan od članova belgijskog Linuks pokreta, odlučio da organizuje manji skup programera softvera otvorenog koda. Nazvao je taj skup *Open Source Developers' European Meeting* (OSDEM). Okupio je prvo tim, zatim su napravili sajt i mejling listu na koju je trebalo da se prijave zainteresovani, međutim, nakon kratkog vremena primetio je da će taj skup biti većih razmera nego što je očekivao. Naredne godine OSDEM je dobio i „F" od „Free" jer nije svaki softver otvorenog koda besplatan, te su organizatori želeli da naglase da ih taj faktor ne zanima.

FOSDEM'17

Brussels / 4 & 5 February 2017

Celo dešavanje smešteno je na briselski univerzitet ULB (fr. *Université Libre de Bruxelles*) koji se sastoji od dvanaest fakulteta i kako naglašavaju - multikulturalnog i internacionalnog tipa, sa programima učenja po najvišim standardima. Same građevine su lepo očuvane i mešavina su modernog i starog kao da je tu nekada bilo nekoliko fakulteta, pa su u međuvremenu dodavali zgrade po potrebi. Za potrebe konferencije na raspolaganju je ove godine bilo pet zgrada fakulteta i par studentskih trpezarija.



Organizacija

Kada na konferenciji koja traje dva dana vidite 645 najavljenih predavanja, jasno vam je da je fizički neizvodljivo da posetite svako koje vas zanima. Možda čak i osetite blago razočaranje kada shvatite da ne možete zapravo da vidite većinu onoga što želite, te vam preostaje da dobro proučite raspored i napravite neku vrstu filtera za ono što biste najviše voleli da slušate, ili nekoga od predavača da vidite. Na sajtu konferencije možete videti kompletan spisak predavanja sa prilično detaljnim opisom, što je dosta bitno jer često iz naslova ne možete precizno da zaključite kakve će se tema zapravo obrađivati. Prostorije su podeljene tematski, što je dobro jer ćete se manje šetati među zgradama i učionicama. Na primer, ako vam je najveće interesovanje Pajton programski jezik, onda ćete sigurno najviše vremena provesti u prostoriji Pajton (*python devroom*) u kojoj je u toku prvog dana bilo dvadeset i četiri predavanja. Pored toga što ima pregršt zanimljivih tema, dodatni problem može biti gužva jer na većinu predavanja morate doći barem deset minuta ranije da zauzmete red, pa nakon što se trenutno predavanje završi i izađe određeni broj posetilaca, počinje novo punjenje prostorije i neretko se desi da čak ni tada nema dovoljno mesta za

sve koji žele da uđu. U takvm situacijama na vrata prostorije volonteri lepe nalepnicu sa obaveštenjem da je prostorija puna, ali takođe i sa *QR* kodom i linkom ka živom prenosu sa konferencije. Naravno IPv6 internet je obezbeđen pa problema sa brzinom interneta nema.

Na info pultu možete dobiti sve potrebne informacije, raspored sa svim predavanjima, informacije u vezi sa prevozom i smeštajima. Međutim taj papir sa spiskom i rasporedom predavanja se nije pokazao kao najbolje rešenje jer kao što sam već napomenuo, postoji 645 najavljenih predavanja, ali se cela lista konstantno ažurira jer se dešava da neko ne može da dođe ili da se čak ubaci još poneko. Organizatori su mislili i na to, pa na *F-Droid* (kao i na Guglovom Plej Storu) možete preuzeti aplikaciju *Fosdem Companion* koja sadrži raspored. Aplikacija takođe ima mogućnost bukmarkovanja što je najbolja stvar koju su mogli da urade jer nakon što proberete kroz raspored i obeležite sve što želite, odete u odeljak *Bookmarks* i vidite vaše odabrane događaje sortirane po vremenu početka. Pored toga aplikacija sadrži i mapu univerziteta što je jako korisno na ovako velikom prostoru, odeljak sa predavačima gde možete pročitati svačiju biografiju. Takođe postoji i odeljak *Live* u kojem možete pratiti predavanje ukoliko niste uspeli da uđete na željeno predavanje ili ukoliko niste otišli na konferenciju.

U četiri od pet zgrada u hodnicima se nalaze štandovi za kojima su predstavnici raznih korporacija, fondacija ili timova programera otvorenog koda koji predstvaljaju neki svoj projekat. Osim „velikih igrača" kao što su CentOS, Fedora, SuSe, Debian, KorOS (eng. *CoreOS*), Postgreskuel (eng. *PostgreSQL*), Majeskuel (eng. *MySQL*), Perl, Pajton, Apači, Libre Ofis, Fondacija slobodnog softvera i ostali, tu su i predstavnici manjih ili čak ličnih projekata. Od manjih projekata izdvajamo RIOT čet više na linku o kojem ćete imati prilike da čitate u LiBRE! magazinu, Olinuksino (eng. OLinuXino) otvoreni hardver, ReaktOS operativni sistem, Vikings slobodni hosting... Svi oni su raspoloženi podjednako da sa vama prozbore po koju reč, daju vam savet ili čak, ako to dozvoljavaju uslovi, odmah reše neki problem.

Predavanja

Volonteri su zaduženi da obaveštavaju predavača tokom celog izlaganja koliko još ima vremena. Pored toga što su predavanja vremenski tačna, zanimljiva je odabir predavača. Naime, ono na šta smo uglavnom navikli jeste da predavač bude osoba koja je dobro proučila određenu oblast ili određeni softver, a zatim prenosi svoja iskustva ili dobre strane neke aplikacije ili servisa. Međutim, na FOSDEM-u je to malo drugačije. Naime, većina predavača su direktni saradnici ili autori. Na primer, u subotu 4. februara je o sigurnoj čet aplikaciji Žici govorio

njen tvorac Saul Ibara Kortege i ispričao kako je pre samo par godina Saul sedeo u istoj toj sali i zapitkivao o bezbednosnim protokolima. Nakon toga, Saul je ukratko objasnio šta je Jitsi jer je većina bila upoznata, a nakon toga uživo instalirao *self hosted jitsi meet* odnosno samostalno servisirani server za aplikaciju (eng. jitsi meet), pokrenuo server dok su se prisutni konektovali i uživo testirali aplikaciju.

Odmah potom su glavnu reč vodili tvorci aplikacije *Converse.js*, a nakon njih programeri koji rade na razvoju aplikacije Ring. Svi oni, osim predstavljanja stoje na raspolaganju za bilo kakva pitanja, vrlo rado razmenjuju kontakte i priželjkuju povratne informacije o svojim aplikacijama i servisima. Nedelja je bila dan naklonjena ljudima koji se bave sajber bezbednošću. U jednom velikom teatru u zgradi Jot i u manjoj sali u zgradi Iks održavala su se predavanja o bezbednosti i enkripciji. Najzanimljivije predavanje iz oblasti bezbednosti bilo je predstavljanje napada (čovek u sredini - eng. *man in the middle* - MITM) na PEAP protokol. Osim samog izlaganja zanimljivo je i to što je predavača Širahei Siniak, studenta četvrte godine tehničkog fakulteta u Belorusiji, najavio čovek koji je pre nešto manje od deset godina radio na razvoju istog protokola. Iz zgrade J izdvajamo izlaganje mladog ali veoma iskusnog i dobrog istraživača iz oblasti sajber bezbednosti - Janson A. Donenfelda koji je predstavio svoj poslednji projekat - VajrGard (eng. *WireGuard*), softverski VPN tunel koji radi vrlo brzo i bezbedno sa najmodernijim algoritmima enkripcije napisan u programskom jeziku C u manje

od četiri hiljade linija koda. Kako je sve besplatno, tako su i snimci (skoro svih) predavanja besplatna i otvorena svima i mogu se naći na sledećem linku.

Iako FOSDEM nije ni prva ni najbolja, a ni poslednja konferencija ovog tipa, svakog puta se može videti nešto novo, ostvariti nova poznanstva, iskoristiti prilika za viđanje sa „starima" ili jednostavno posetiti Brisel i pretvoriti u još jednu avanturu kroz svet hardvera i softvera otvorenog koda u društvu ljudi otvorenog srca.

DORS/CLUC Konferencija

Autor: Ivana Isadora Devčić

Prateći izvještaje sa FLOSS konferencija i postove na društvenim mrežama lako je dobiti dojam da se sve velike i važne stvari događaju negdje daleko. Ipak, postoji manifestacija koja već godinama privlači najveća imena zajednice slobodnog i otvorenog softvera, a nećete morati putovati na drugi kontinent da biste ju posjetili. Riječ je o DORS/CLUC konferenciji, koja se održava u glavnom gradu Hrvatske, i koja je prema nekim izvorima najstarije i najveće takvo okupljanje u regiji.

Priča počinje sada već davne 1993. godine, kada su održani prvi Dani otvorenih računarskih sustava (DORS) u organizaciji „Hrvatske udruge za otvorene sustave i Internet" (HrOpen). Tijekom godina DORS je iz godišnjeg okupljanja nekolicine entuzijasta prerastao u centralni događaj FLOSS zajednice u Hrvatskoj, ali i šire. Od 1999. održava se zajedno s Hrvatskom konferencijom Linuks korisnika (eng. CLUC - Croatian Linux Users Conference), kada je i nastalo današnje ime konferencije - DORS/CLUC.

Konferenciju zajednički organiziraju dvije neprofitne udruge, HrOpen i HULK (Hrvatska udruga Linuks korisnika) uz potporu sponzora, a već niz godina i uz podršku Hrvatske gospodarske komore. Obje udruge svojim radom značajno doprinose i utječu na svijest o važnosti slobodnog i otvorenog softvera u Hrvatskoj.

Uz nekoliko iznimaka, konferencija se sada već tradicionalno održava na

Predstavljamo

Fakultetu elektrotehnike i računarstva (FER) u Zagrebu, te obično traje tri dana. Prva dva namijenjena su predavanjima, dok se treći dan održavaju praktične radionice.

Predavanja obuhvaćaju širok spektar tema: od poslovnih primjena otvorenih tehnologija i implementacije slobodnog softvera u institucijama do prezentacija najnovijih hardverskih dostignuća, novih FLOSS projekata, i savjeta za učinkovitiji razvoj i korištenje softvera. Osim predavanja, DORS/CLUC organizira i tribine, odnosno rasprave otvorenog tipa, od kojih valja istaknuti Žene i svetu FLOSS tehnologija (eng. *Women in Floss technology world*) iz 2014.

Uz domaće predavače, na DORS/CLUC dolazi i velik broj predavača iz europskih zemalja (Njemačke, Francuske, Mađarske, Italije, Bugarske, Srbije, Albanije, Austrije...), pa čak i iz SAD-a. Predstavnici OpenForuma Europe, Ken Kor (Apače fondacija softvera), Mark Šatlvort (Kanonikal) i Matijas Kiršner (FSFE) samo su neke od poznatih figura FLOSS zajednice koje su se predstavile na DORS/CLUC pozornici tijekom godina.

Dors/Cluc Konferencija

DORS/CLUC je konferencija otvorena svima - studentima, korisnicima Linuksa, razvijateljima softvera, hakerima, malim i srednjim poduzetnicima, velikim tvrtkama, ali i predstavnicima javnog sektora - koji su zainteresirani za razvoj i promicanje kulture slobodnog i otvorenog softvera. U pauzama između predavanja posjetitelji mogu razgledati štandove izlagača, vidjeti demonstracije rada raznih uređaja poput Rozberi Pi računala i 3D printera, ali i doznati više o lokalnim udrugama te velikim projektima i organizacijama kao što su Mozila i Fedora.

Predstavljamo

Posjetitelji DORS/CLUC opisuju kao „mini FOSDEM", ali i kao konferenciju na kojoj vlada izrazito prijateljska atmosfera. Istina, predavanja se bave ozbiljnim temama, i posjetitelji razmjenjuju poslovna iskustva, savjete i kontakte, ali jednako tako se na konferenciji dogovaraju i buduća neformalna druženja te razmjenjuju naljepnice i ostali konferencijski promotivni materijal. Nađe se tu i palačinki, zatim uvijek popularnih lebdećih pingvina, a nakon završetka konferencije redovito se održava i veliki parti.

Prošlogodišnja konferencija bila je posebna po tome što je po prvi put bilo moguće pratiti predavanja putem uživo prenosa (eng. Livestream). Tijekom 3 dana predavanja i radionica prenos uživo je pratilo 1200 virtualnih posjetitelja, što također predstavlja svojevrstan rekord.

Dors/Cluc Konferencija

Ove godine očekuje se više stotina posjetitelja na lokaciji koja će ponovno biti FER u Zagrebu, na adresi Unska 3. Javnim prijevozom se jednostavno i relativno brzo može doći do lokacije od glavnog željezničkog i autobusnog kolodvora, a u blizini je i nekoliko opcija za smještaj koje posjetiteljima konferencije nude poseban popust. DORS/CLUC 2017 održat će se od 31. svibnja do 2. lipnja, a ulaznice po povoljnijoj cijeni dostupne su do 15. svibnja.

Potpuni raspored predavača bit će objavljen uskoro, no već sada je sigurno da će DORS/CLUC 2017 obilježiti brojne korisne i živopisne prezentacije. Heiki Lohmus (FSFE) govorit će o primjeni slobodnog softvera u STEM obrazovanju, dok će Marian Marinov održati čak 3 predavanja. Sve novosti o DORS/CLUC 2017 možete pratiti na službenom sajtu konferencije (https://2017.dorscluc.org/) te na društvenim mrežama. Ako dosad još niste, nadamo se da ćete ove godine posjetiti DORS/CLUC, te vas s veseljem očekujemo!

Predstavljamo

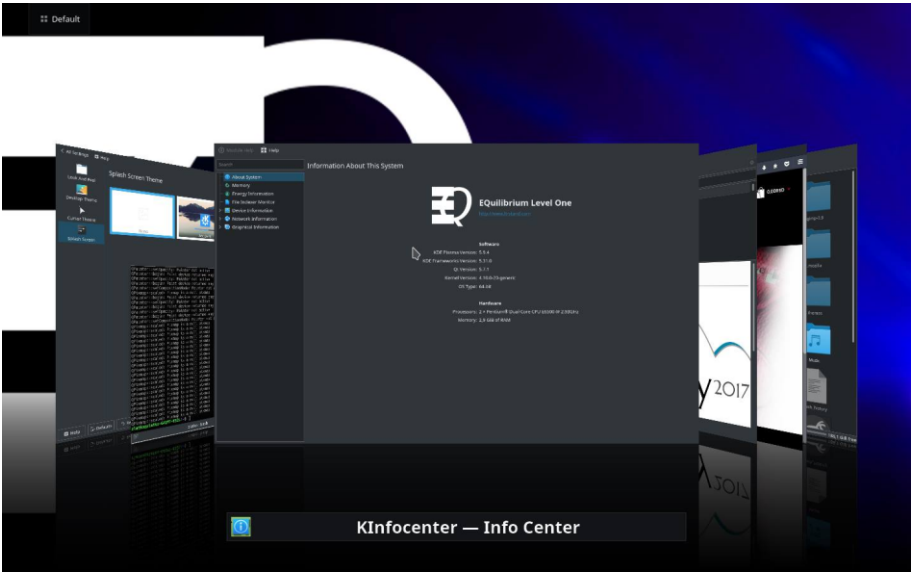
Ekvilibrijum

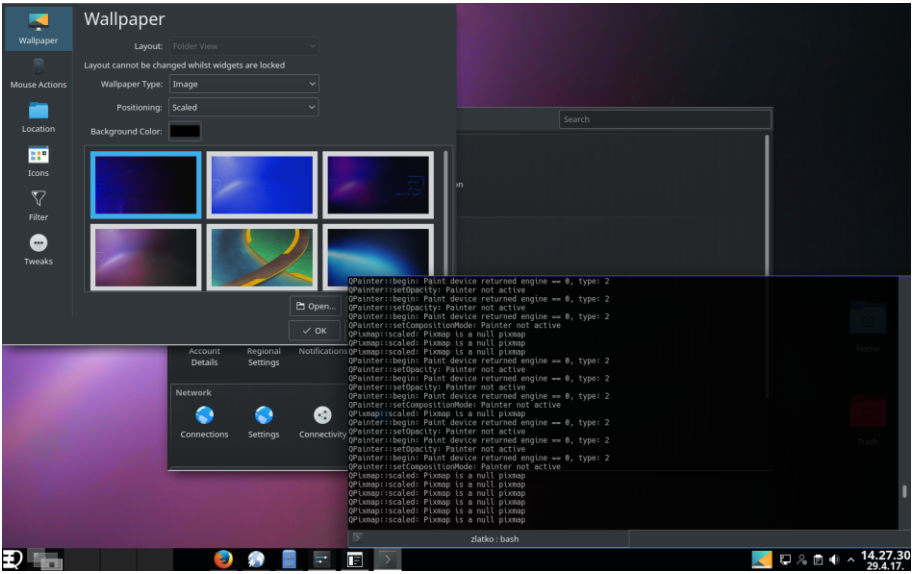
Autori: Zlatko Barnak i Nikola Todorović

Predstavljamo vam prvo izdanje GNU/Linuks distribucije pod nazivom Ekvilibrijum „Nivo Jedan" (eng. *EQuilibrium „Level One"*). Kao što i sam naziv kaže, filozofija kojom se Ekvilibrijum vodi je zapravo ravnoteža i spoj različitih okruženja, čineći time prepoznatljivo ali nešto drugačije korisničko iskustvo. Pomalo jedinstven spoj rezultirao je brzom, modernom i elegantnom distribucijom.

Ekvilibrijum je nekim svojim karakteristikama više okrenut prenosnim računarima (laptop, ultrabuk i sl.), ali to ne znači da su desktop računari okruženje u kojem se lošije snalazi - naprotiv.

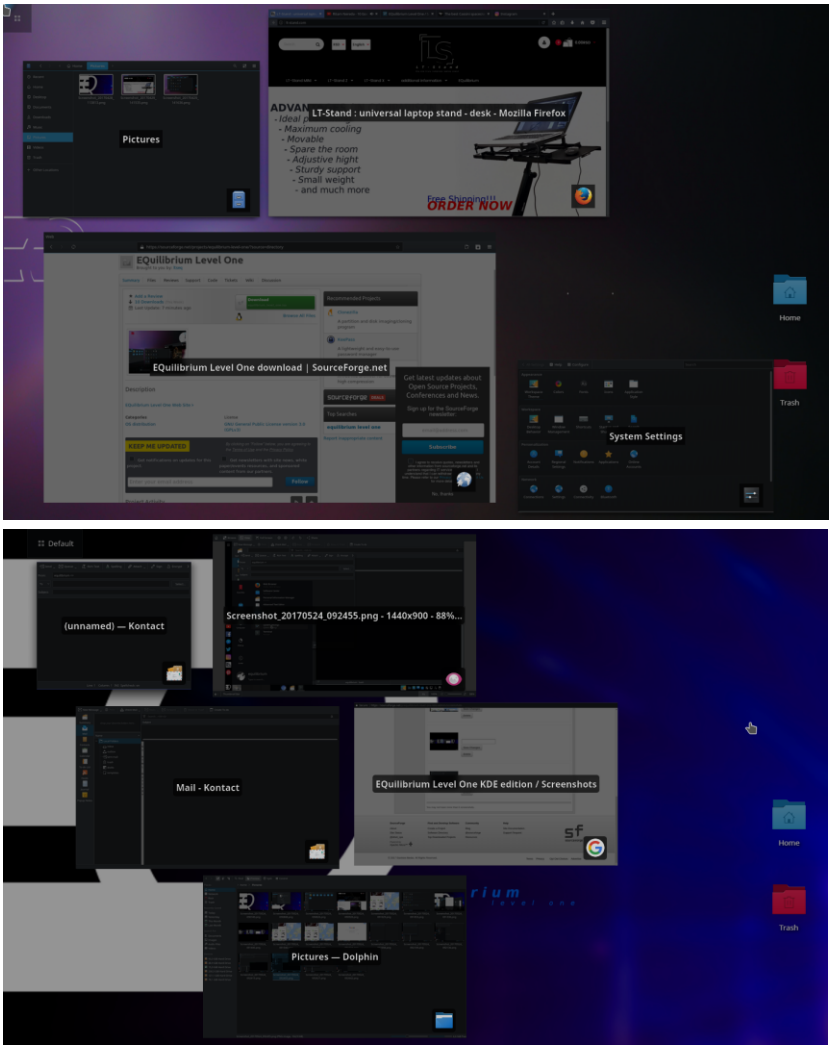
Ekvilibrijum donosi pomalo izmenjeno okruženje u kojem su nazivi programa i dugmići izmešteni izvan samog prozora u gornji desni ugao radne površine. Na prvu loptu to deluje kao hendikep, ali posle nekoliko minuta privikavanja vidi se prednost ovakvog okruženja, pogotovo na manjim monitorima. S ovakvim uređenjem postignuta je uniformisanost i sami prozori deluju da su deo celine sa radnom površinom, a ne zasebni programi. Ciljna grupa za koje je ovaj operativni sistem namenjen su manje zahtevni korisnici, koji nemaju potrebu da svaki čas nešto menjaju ili podešavaju, i ljudi kojima je potreban upotrebljiv i stabilan operativni sistem (žargonski rečeno). Programi se otvaraju jako brzo, a kretanje između programa je munjevito i cela distribucija radi jako stabilno i pouzdano.


Default
Information About This System
EQuilibrium Level One
KInfocenter — Info Center

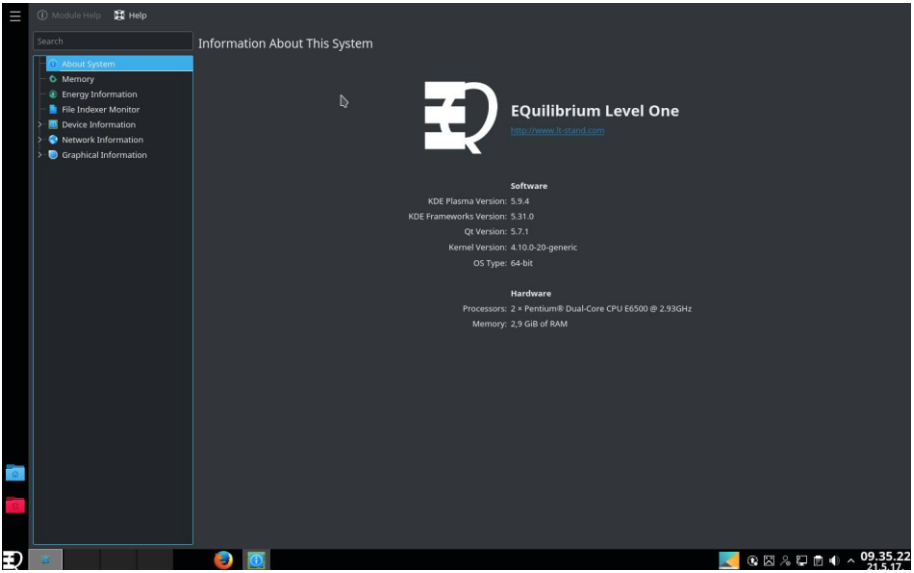

Wallpaper
Layout:
Folder View
Layout cannot be changed whilst widgets are locked
Wallpaper Type:
Image
Positioning:
Scaled
Background Color:
Mouse Actions
Location
Icons
Filter
Tweaks
Open...
OK
Account Details
Regional Settings
Notifications
Network
Connections
Settings
Connectivity
zlatko : bash

Predstavljamo

Ova distribucija donosi sve što je potrebno da pokrenete vašu organizaciju, školu, kuću ili preduzeće. Ispod haube se nalazi 4.10 verzija Linuks kernela. Sve bitne aplikacije su uključene, pa ćete se tako susreti sa poznatim programom za upravljanje datotekama Nautilusom, uređivačem slika pod nazivom GIMP. Takođe su prisutni i VLC video-plejer, Kontakt, Fajerfoks, Libre Ofis i mnogi drugi.



Distribucija deluje vrlo lagano i poletno čak i na slabijim računarima. U samom startu teži oko 400 megabajta radne memorije(RAM), ali radi nesmetanog funkcionisanja poželjno je pokretati je na računaru koji ima bar 3 gigabajta raadne memorije, dvojezgrarni procesor od dva ili više gigaherca i minimum 16 gigabajta slobodnog prostora na hard disku.

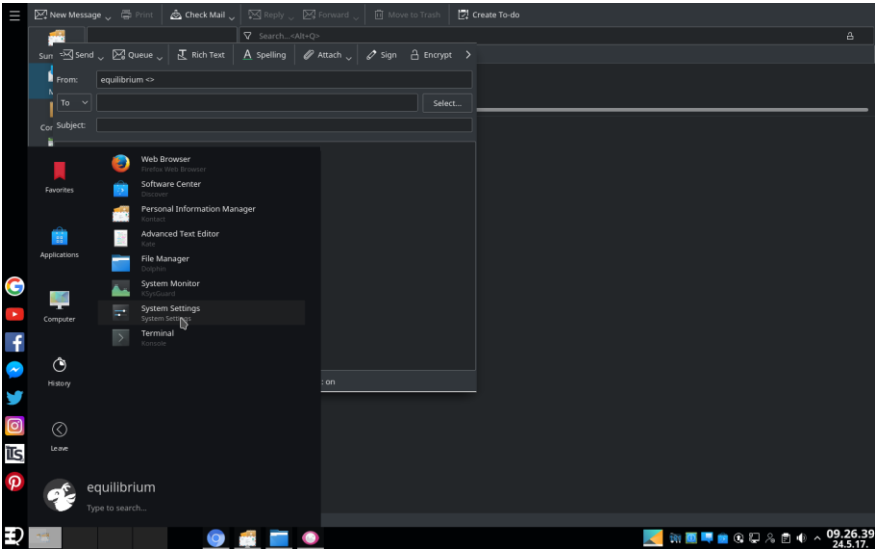


Autor ovog članka, a ujedno i tvorac ove distribucije (Zlatko Barnak), koristi Ekvilibrijum na više računara i do sada se nije susretao sa većim problemima u radu distribucije. Iz tog razloga poziva čitaoce LiBRE! časopisa da isprobaju distribuciju i iznesu svoje utiske, jer kao i kod svakog projekta slobodnog softvera, odziv zajednice predstavlja jedan od najvažnijih delova u razvitku projekta, u ovom slučaju operativnog sistema.

Ukratko rečeno, Ekvilibrijum predstavlja modernu 64 bit-nu distribuciju, temeljenu na Kubuntuu, čiji video pregled možete videti na jutjubu kao i uputstvo za podešavanja nakon instalacije.

Predstavljamo

Ekvilibrijum “Nivo Jedan” je dostupan za preuzimanje na sledećem linku, a nedavno se pojavila i KDE verzija Ekvilibrijuma koja je dostupna za preuzimanje na ovom linku, a takođe možete posetiti i zvaničnu stranicu distribucije.



Način povratka izgubljenih podataka - Fotorek

Autor: Milan Stojakov

Slučajno obrisani dokumenti ili draga slika danas je veoma česta pojava što zbog brzine kojom upravljamo podacima, što zbog programerskih grešaka odnosno bagova, a vrlo često i usled otkazivanja uređaja. Nikada se ovaj problem ne može u potpunosti predupredi ti pa je zato dobro da znamo šta možemo učiniti u slučaju gubitka bitne informacije.

Fotorek

Fotorek (eng. *Photorec*) je slobodan i besplatan alat, nastao od Testdisk (eng. *Testdisk*) programa. Njegov opseg funkcija je znatno umanjen u odnosu na Testdisk, što je u ovom slučaju dobro i smatramo da je jedan od najboljih alata kada je potrebno hitno povratiti izgubljenu datoteku. Fotorek pristupa nosačima podataka u režimu čitanja i time ne nanosi dodatnu štetu izgubljenim podacima. Fotorek spasava izgubljene slike, pdf dokumente, video-zapise i tome slično. Za njegovo korišćenje nije potrebno poznavati dublju problematiku upravljanja datotekama, jer je njegov interfejs prilično jednostavan za korišćenje. Fotorek je dostupan za sve popularne operativne sistem što uključuje vindouz, linuks i mek os.

Kako da...?

Nestale slike

Prikazaćemo vam spasavanje podataka sa Es-De kartice koja se koristila u telefonu i gde su podaci nestali usled korupcije kartice ili softverske greške. Nema veće suštinske razlike između spasavanja podataka sa memorijskih kartica, hard diskova i sličnih medijuma, ali ćemo se zarad jednostavnosti teksta usredsrediti na memorijske kartice. Obratite pažnju na koji način ćete povezati memorijsku karticu na računar, jer čitač igra bitnu ulogu u brzini upisa i čitanja. To će znatno uticati na vreme završetka posla koji može da traje ili jedan čas ili jedan dan. Kada slučajno obrišete datoteku, najvažnije je da više ne koristite odabrani medijum odnosno memorijsku karticu, zbog samog načina funkcionisanja kartice: svakim daljim upisom novih podataka smanjuje se šansa za vraćanje obrisane datoteke. **Napominjemo** da potpuni oporavk izgubljenih datoteka nije uvek moguć i zavisi od mnogih faktora. Zbog samog načina funkcionisanja sistema datoteka, jako je bitno da kada uočite da vam nedostaju tražene datoteke - medijum stavite van funkcije. Datoteke nikada nisu u potpunosti izbrisane, jer je tako projektovan sistem podataka, da bi se povećala brzina rada, a za samog korisnika nije od velikog značaja da li je datoteka fizički prisutna na hard disku ili ne. Datoteke koje su obrisane, ili koji su nestale, nisu fizički obrisane sa medijuma, već se njihov indeks briše iz tabele podataka, što znači da datoteka i dalje postoji fizički na medijumu, ali se sistem ponaša kao da je obrisana i biće uništena tek kada se na njeno mesto upišu novi podaci. Zbog toga nije uvek moguće izvršiti potpuno spasavanje podataka, pogotovo ako se medijum aktivno koristi jer velike su šanse da su prazna mesta gde su se nekada nalazili podaci, popunjena i time prethodni podaci uništeni.

Instalacija

Program možete preuzeti ovde, ili instlalaciju možete obaviti korišćenjem upravnika paketa distribucije, jer mnoge linuks distribucije već imaju Fotorek u svojim riznicama.

Ubuntu - Mint:

```
sudo apt install testdisk
```

Fedora:

```
sudo dnf install testdisk
```

Arč:

```
pacman -s testdisk
```

Paket sa sajta ne zahteva instalaciju već je dovoljno pokretanje putem terminala. Njegovo pokretanje se vrši na sledeći način: Otvorite terminal, korišćenjem komande cd pronađete direktorijum u kojoj je raspakovana arhiva i pokrenete Fotorek korišćenjem komande

```
sudo ./photorec
```

Ako je instaliran iz zvaničnih riznica distribucije sledećom naredbom

```
 sudo photorec
```

Spasavanje podataka

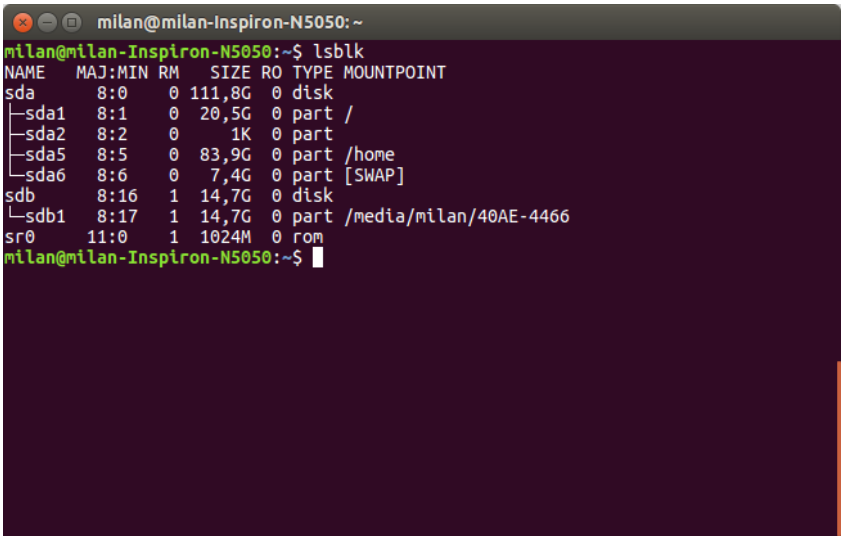
Princip upotrebe programa i okruženje je identično između Linuks i Vindouz verzije, zato ono što je ovde prikazano važi za obe platforme. Fotorek izvlači sve obrisane podatke koje uspeva da nađe. To znači da se ne bira jedan deo gde ćemo vršiti oporavak, već se cela particija pretražuje. Upravo zbog toga je bitno da imamo dovoljno slobodnog prostora na odredišnoj lokaciji, najmanje dvostruka veličina kapaciteta od same particije. Moguće je postaviti određeni tip datoteke koje želimo da spasemo tj. prema određenoj ekstenziji, ako želimo tačno određen tip datoteke da spasemo. Fotorek prepoznaje preko 480 različitih ekstenzija datoteka. Kretanje kroz program vršimo na sledeći način: tasterom **enter** vršimo odabir, a tasterom **Q** vraćamo se nazad.

Kada smo upoznati sa mogućnostima i načinom rada programa, možemo početi sa radom.

1. korak

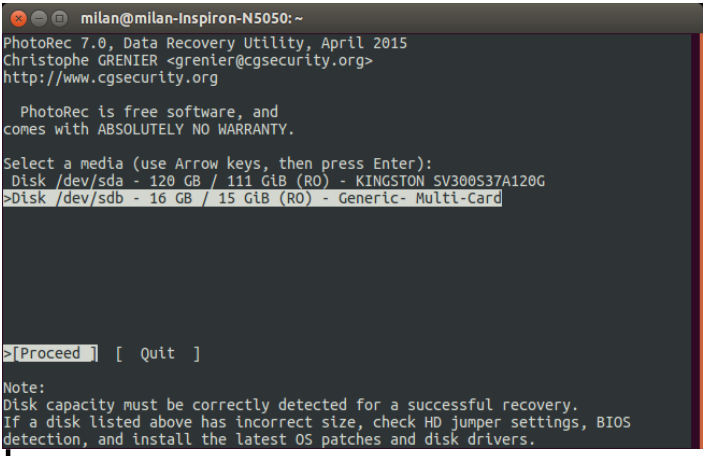
Otvorimo terminal i ispišemo listu povezanih medijuma na računar putem komande:

```
lsblk
```



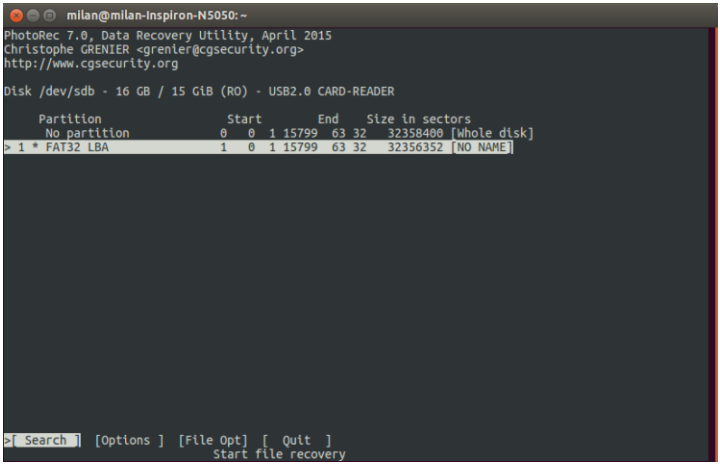
Informacija o oznaci medijum je jako bitna i predlažemo da je zapišete negde kako je ne biste zaboravili. Kao što se može videti, naša kartica je obeležena sa **dev/sdb**. Nju možemo prepoznati po kapacitetu koji je ovde 14,7 gigabajta. Obratite pažnju da je to stvarni kapacitet kartice, a ne onaj koji piše na samoj kartici. Ova kartica je deklarisana na 16 gigabajta.

Podstavka sa oznakom **dev/sdb1** označava particiju koja se nalazi na uređaju **dev/sdb**. Trenutno nam je bitan uređaj sa podacima za spasavanje čija je oznaka u ovom slučaju **dev/sdb**, što ćemo i zapisati sa strane na papir kako je ne bismo zaboravili. Nakon pokretanja biramo uređaj sa kojeg želimo da vršimo spasavanje podataka, u ovom slučaju **dev/sdb** uređaj. Veoma je bitno da smo sigurni da je urađaj koji je ovde izabran onaj uređaj sa kojeg vršimo spasavanje podataka. Najlakše je identifikovati karticu njenim kapacitetom, koji je prikazan sa desne strane ili ako ste prihvatili naš savet i zapisali oznaku uređaja. Kapacitet je prikazan u formatu nominalni (onaj koji piše na kartici)/ stvarni (onaj koji je zaista na kartici).



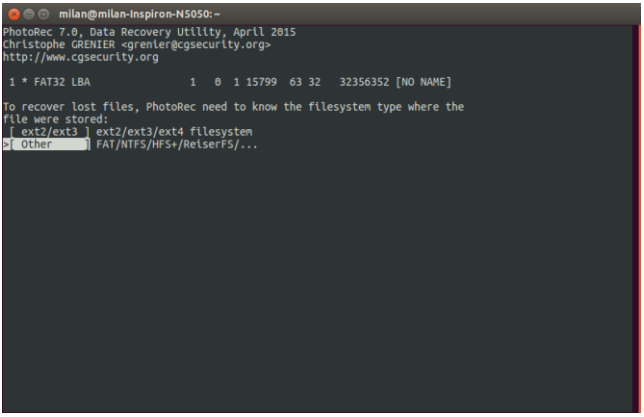
2. korak

Sledeća opcija je prostor koji skeniramo tj. particiju. Particija je obeležena formatom *FAT32*. Format memorijske kartice u većini slučajeva je *FAT32*. Druga opcija je da se skenira neparticionisani deo, tj. deo koji nije formatiran. Memorijske kartice tipično imaju jednu particiju, dok hard disk uređaji obično imaju više od jedne particije. Kako je ovde memorijska kartica u pitanju, možemo videti samo jednu particiju. Particija ne mora da zauzima sav dostupan prostor, takođe može se desiti da particija nestane što je isto čest kvar, ali se u ovom tekstu nećemo baviti time.



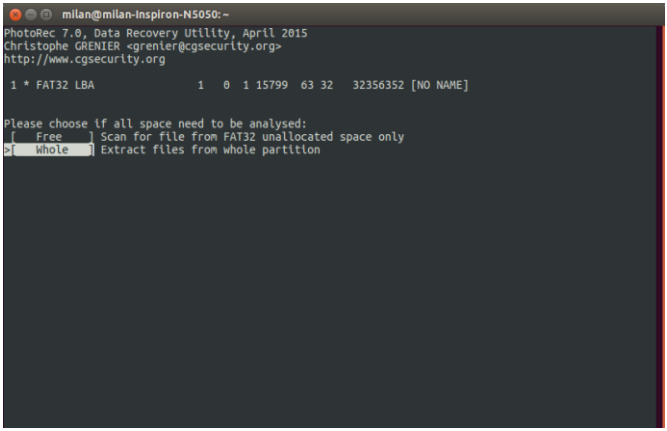
3. korak

Format particije. Ova informacija je dostupna gore u nazivu diska. Ova opcija je korisna jer postoje i slučajevi kada je oštećenje particije veliko i program ne može sa sigurnošću utvrditi postojeći format.



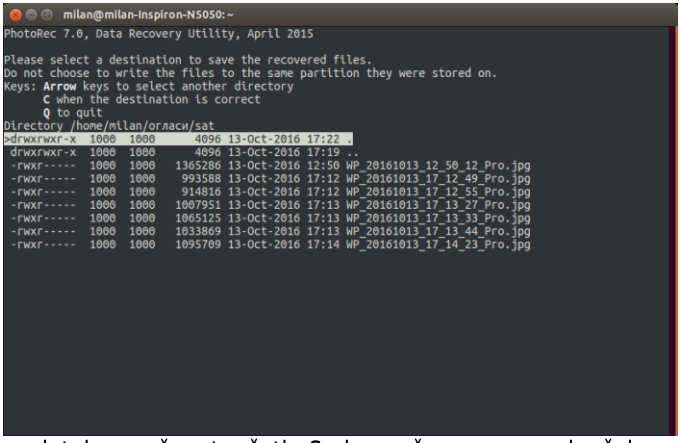
4. korak

Odabir prostora za analizu je bitna stavka kojom određujemo koji prostor na memorijskoj kartici ćemo pretraživati. Opcijom **Free** pretražuje se samo trenutno prazan prostor na memorijskoj kartici, dosta brže od opcije **Whole**. Ova opcija je korisina u slučaju kada je datoteka skoro obrisana. Opcija **Whole** je mnogo temeljnija, ali zahteva dosta više vremena jer se ceo disk pretražuje.

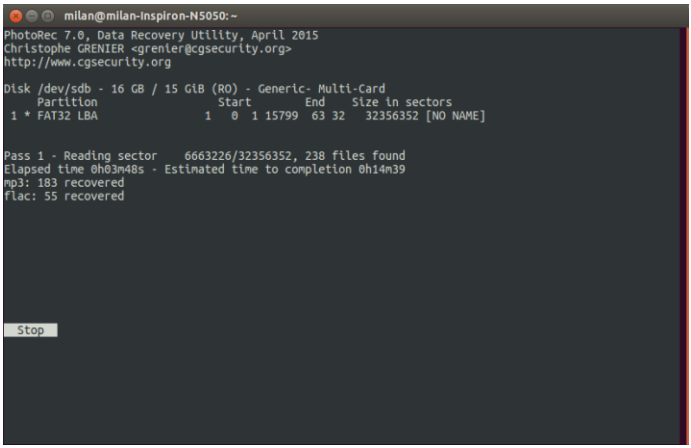


5. korak

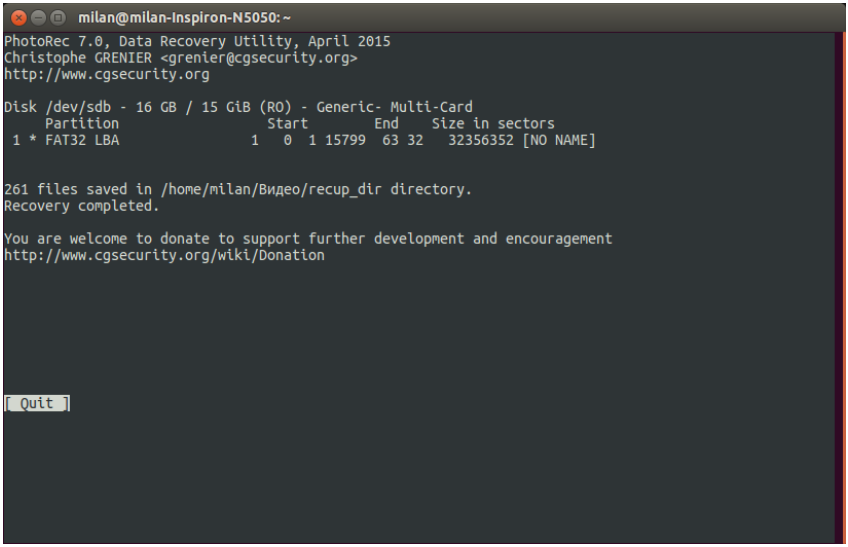
U ovom korak biramo lokaciju gde će se čuvati spaseni podaci. Nemojte nikada birati da se podaci čuvaju na onom disku odakle se izvlače podaci, jer će se time uništiti podaci koje pokušavate da oporavite. Pobrinite se da na hard disku ima dovoljno mesta za oporavak podataka, najmanje dvostruka vrednost kapaciteta particije odnosno kartice. Fotorek odabir lokacije za čuvanje podataka otpočinje u ličnoj fascikli korisnika koji vrši spasavanje podataka odnosno **/home** direktorijumu.



Oporavak podataka može otpočeti. Sada možemo samo da čekamo i da se nadamo najboljem.



U našem slučaju, ukupno smo imali 251 oporavljenu datoteku, koje je sada potrebno pregledati i izvući ono što nam je korisno. Primetite da datoteke imaju nasumične nazive, a ne stare originalne nazive.



Spasene datoteke

Datoteke će se nalaziti u direktorijumima sa nasumičnim nazivima npp **./recup_dir.1/f4586900444.jpg**, **./recup_dir.2/f874546789795.jpg** Male sličice predpregleda počinju sa slovom **t** npr. **t84984614.jpg**. To su jako male sličice od kojih generalno nemamo koristi.
Mi smo u našem slučaju imali memorijsku karticu na kojoj su se nalazile muzičke datoteke. Bez problema smo uspeli da izvršimo povraćaj svih izgubljenih datoteka.

U sledećem delu ćemo opisati kako da korišćenjem alata Testdisk napravite odraz medijuma, što samu proceduru spasavanja podataka čini mnogo efikasnijom, fleksibilnijom i što je najvažnije bezbednijom. Osvrnućemo se i na skriptu za sortiranje spasenih podataka.

Prvi koraci u Blenderu (1. dio)

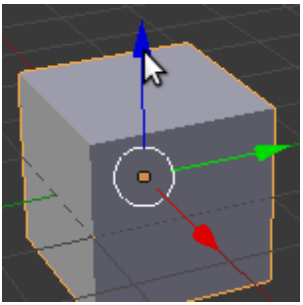
Autor: Slobodan Vidović

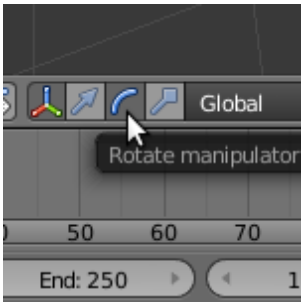
Prije nego se upustimo u 3D svijet sa Blenderom, hajde da vidimo kako je sve počelo, ko su autori i kako je Blender postao otvorenog koda. Prvobitni kreatori Blendera bili su holandski studio Neo Geo i NaN, a nosio je potpis autora Tona Rosendala (eng. *Ton Roosendaal*) koji je, ispostaviće se, i danas odgovoran za razvoj ovog programa. Rosendal je osnovao NaN u junu 1998. godine kako bi osigurao dalji razvoj Blendera. Naziv Blender je inspirisan pjesmom „Jelou" sa albuma „Bejbi". Tada je Blender distribuiran pod šerver (eng. *shareware*) licencom, međutim NaN je bankrotirao 2002. godine i kreatori su se dogovorili da Blender objave pod GNU-ovom opštom javnom licencom pod uslovom da kompaniji Neo Geo bude plaćeno 100 000€. Tako je 18. jula 2002 god. Rosendal osnovao fondaciju Blender i započeo kampanju za prikupljanje sredstava koja je trajala do 7. septembara 2002. kad je objavljeno da je sakupljeno dovoljno sredstava i da će kod programa Blender biti objavljen. Blender je tada postao slobodan softver kakvim ga i danas znamo. Blender je dostupan u 32-bitnoj kao i 64-bitnoj verzija za linuks, friBSD i vindouz, napisan je u programskom jeziku Pajton, trenutna stabilna verzija Blendera je 2.78. Kako bi se osigurao dobar i brz razvoj Blendera, najistaknutiji Blender umijetnici i programeri su se okupili seembra 2005. godine i započeli Oranž muvi projekat (eng. *Orange Movie*) koji je za cilj imao kreiranje kratkog animiranog filma koji je kasnije dobio naziv „Elefant drim" (eng. *Elephant Dream*) i premijerno prikazan 24. marta 2006. Zbog velikog uspjeha tog kratkog animiranog filma, Blender fondacija je osnovala Blender institut kako bi svake godine okupljali Blender umjetnike i programere kao i da bi radili na novom kratkometražnom animiranom filmu koji kao rezultat imao dodavanje novih alata u Blenderu uporedo sa razvojem filma.

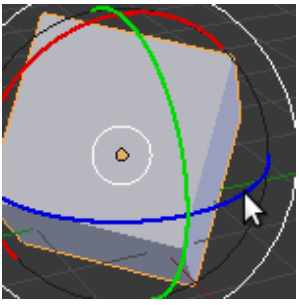
Kako da...?

Nažalost, nema mnogo upustava za Blender na našem jeziku, zbog toga ćemo se u daljem tekstu pokazati kako početi koristiti Blender. Nakon pokretanja programa, dočekaće vas specifično grafičko okruženje koje možete po želji vrlo lako da prilagodite sopstvenim potrebama. Prvo što vidite jeste više prozora čije dimenzije možete po želji da promijenite, centralni odnosno 3D prikaz prozor je najveći i u njemu su prikazani 3D objekti u vašoj sceni, ispod njega se nalazi vremenska linija (eng. *timeline*), a iznad njega nalazi se prozor sa informacijama, prozor koji nam prikazuje najosnovnije informacije. Sa desne strane nalazi se od vrha prema dnu autlajner (eng. *outliner*) koji nam prikazuje listu objekata u našoj sceni, ispod njega nalazi se prozor sa svojstvima (eng. *properties*) koji se sastoji iz više sekcija a to su: render (eng. *render*), scena, svjet, objekat, objektova ograničenja (eng. *object constraints*), izmijenjivači objekata (eng. *object modifiers*), podaci o objektu, materijal, teksture, fizika.

Hajde da se fokusiramo na centralni 3D prozor u kom možemo da vršimo manipulaciju nad 3D objektima. Kako bismo selektovali podrazumijevanu 3D kocku, dovoljno je da stavimo miš na nju i da pritisnemo desni klik, da desni klik miša što je specifična funkcija za blender da se selektovanje vrši desnim klikom, a ukoliko pritisnemo lijevi klik promijenićemo poziciju 3D kursora koji je jako bitan, a koristi se za na primer rotiranje objekta na osnovu pozicije 3D kursora. Ukoliko dodamo novi objekat u našu scenu, on će se pojaviti na mjestu gdje se nalazi 3D kursor. Ako želimo da promijenimo pozciju naše kocke, to možemo da uradimo na dva načina: prvi je da pritisnemo taster **g** *grab* i pomijerimo miš lijevo ili desno i kocka će pratiti putanju miša. Ovaj način pomijeranja objekta je slobodan, dakle možemo objekat da postavimo na bilo koju poziciju. Ukoliko pritisnemo **x**,**y** ili **z** na tastaturi dobićemo opciju da selektovani objekat pomijeramo po odabranoj osi što je i podrazumjevano pomijeranje na drugi način, odnosno da strelicom našeg miša pritisnemo plavu, zelenu ili crvenu strelicu koja polazi iz centralne tačke naše kocke (pogledaj na slici 3), zadržimo klik i pomijerimo miš u pravcu pokazivanja strelice. Primijetićete da se kocka kreće samo po označenoj osi, u zavisnosti od toga da li ste pritisnuli na crvenu, zelenu ili plavu strelicu. Rotaciju kocke, takođe, možemo da izvršimo na dva načina: prvi je pritiskom na taster **r** *rotate*, nakon toga pomijerimo miš i kocka će se rotirati za određeni broj stepeni, a ukoliko unesemo neku numeričku vrijednost na tastaturi, recimo 45, kocka će se rotirati za 45 stepeni. Drugi način za rotiranja je promijenom dugmeta za prikaz tipa manipulacije na kocki (slika 4) koji će promijeniti strelice za pomijeranje u kružne objekte za rotaciju (slika 5).

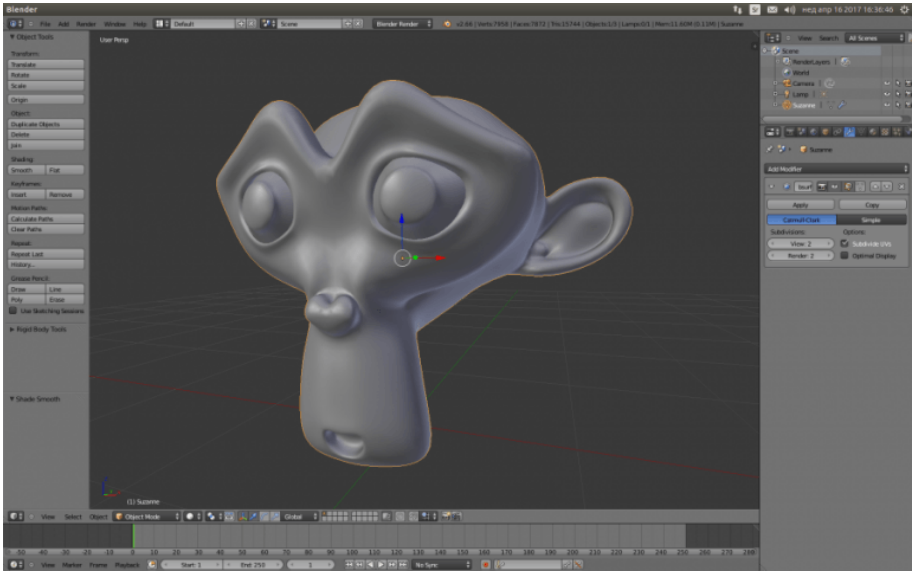






Uvećavaanje ili umanjenje kocke možemo da izvršimo na dva načina: prvi je pritiskom na taster **s** *scale* i nakon toga pomijerimo miš naprijed ili nazad, a kocka će se uvećati ili umanjiti, drugi način je da promijenimo dugme za prikaz tipa manipulacije (slika 4) odabirom opcije 3. Nakon izvršenja pomijeranja, rotacije ili uvećanja kocke da potvrdimo odnosno da sačuvamo trenutno stanje

potrebno je da pritisnemo lijevi klik miša. Ukoliko pritisnemo desni klik, manipulacija nad našim objektom će biti otkazana i vratiće se u prvobitno stanje. Sada ćemo da obrišemo kocku pritiskom na taster **x** i potvrdom na lijevi klik miša, zatim ćemo na numeričkoj tastaturi da pritisnemo taster **1** kako bismo prešli u **Front Persp** pogled, zatim ćemo da pritisnemo **Shift + A**, pa idemo na **mesh**, pa na **Monkey**. Objekat prikazan na ekranu je maskota Blendera koja se zove Suzan. Kako bismo pogledali Suzan sa lijeve strane, treba da pritisnemo **3** na numeričkoj tastaturi. Da bismo pogledali Suzan sa prednje strane, treba da pritisnemo **1** na numeričkoj tastaturi.

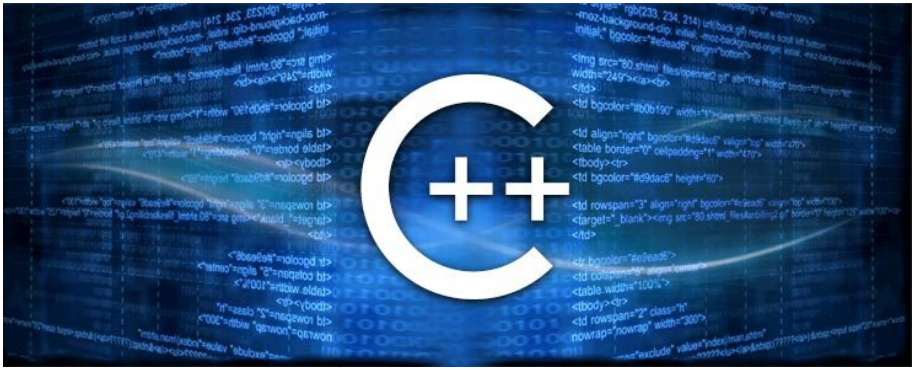


Tu ćemo da završimo naše putovanje sa upoznavanjem Blendera u ovom dijelu, u sledećem nastavku ostaje nam da se upoznamo sa napredim opcijama koje nam nudi Blender. Do slijedećeg čitanja, srdačan pozdrav.

Uvod u objektno orjentisano programiranje C++ (1. dio)

Autor: Amar Tufo

U ovom stručnom članku bit će predstavljen pojam objektno orjentisanog programiranja kreiranjem praktičnog programa, te upotreba klasa i objekata kao jednog od osnovnih novina objektno orjentisanog programiranja u C++ programskom jeziku. Osnovni cilj je da se nešto iskusniji hobisti, studenti i programeri upoznaju sa načinom kreiranja klase u zasebnoj datoteci poznatoj kao zaglavlje, njeno pozivanje u programu, kreiranje objekta, te upotreba mutatora i aksesora kako bi se postavile vrijednosti određenim tipovima podataka. Ovaj stručni članak koji je nastao u potpunosti na Ubuntu linuks distribuciji kao platformi, demonstrira upotrebu Emaksa kao tekst editora, te kompajliranje C++ datoteka upotrebom GNU/G++ kompajlera. Cio završni projekat je dostupan na mom githabu, pa ukoliko ste upoznati sa upotrebom Githaba, ovaj projekat možete klonirati na svoje računalo kako biste ga proširili, izveli modifikacije te dodali neke nove dodatke.



Kako da...?

Kratak uvod u objektno orjentisano programiranje

Prije nego li krenemo u detalje razrađivati ovaj članak, imam tu potrebu da najprije nešto kažemo o objektno orjentisamom programiranju, zašto je ono važno za C++ programski jezik, koje su prednosti i karakteristike ovog načina razvoja softvera itd. Kada se pogleda na današnji softver, možemo slobodno reći da je on dostupan ne samo na Desktop platformama, nego i mobilnim, pa i u klaudu. Ovo možemo zahvaliti prvenstveno novim programskim jezicima kao što su PHP, Pajton, Java, C++, VB.NET, Rubi On Reils i mnogi drugi programski jezici koji su omogućili lako portanje softvera, ne samo na druge platforme nego i u druge programske jezike. Nije uvijek tako bilo. Naime, 1960-ih godina, programeri su se našli u velikim problemima razvoja softvera. Velik broj ljudi se angažovao na razvoju jednog projekta koji je naravno trebalo da ubrza proces njegovog razvoja, ali nije. Softver je postao skup ne samo za razvoj nego i njegovo održavanje, postao je nestabilan i bagovit i povrh svega, krajnji korisnici su bili nezadovoljni ali i zahtjevni. Ovo je kasnije postalo poznato u informatici kao **softverska kriza**. Programeri su tragali za rješenjem ovog problema, te se kao odgovor na softversku krizu javlja objektno orjentisano programiranje. Dakle, objektno orjentisano programiranje, OOP nastaje kao odgovor na tzv. softversku krizu koju su izazvali sami programeri. Oni su ljudima pokazali što sve jedan računarmože,

puno više od onoga što su mislili i oni kao tvorci softvera, te krajnji korisnici njihovog softvera. Međutim, OOP uvodi svojevrsnu revoluciju u načinu izrade softvera u čijem primarnom fokusu se nalazi objekat. Stoga, OOP te C++ kao objektno orjentirani programski jezik u razvoj softvera unose ova rješenja:

Apstrakcija tipova podataka (eng. *Apstract Data Tupes*) – kao što u C-u ili u nekom drugom programskom jeziku postoje ugrađeni tipovi podataka (intidžer, karakter ...), tako u jeziku C++ korisnik može proizvoljno definisati svoje tipove podataka (klase), te ih ravnopravno koristiti tijekom razvoja svog softvera.

Enkapsulacija (eng. *encapsulation*) – realizacija nekog tipa može (i treba) da se sakrije od ostatka sistema (od onih koji ga koriste). Treba korisnicima tipa precizno definisati samo šta se sa tipom može raditi, način kako se to radi, odnosno sakriva, definiše se interno.

Nasljeđivanje (eng. *inheritance*) – pretpostavimo da je već formiran tip printer koji ima operacije *print_line*, *line_feed*, *form_feed*, *goto_xy* itd, te da je njegovim korištenjem već realizovana velika količina softvera. Novost je da je firma nabavila i štampače koji imaju bogat skup stilova pisma i želja je da se oni ubuduće iskoriste. Stoga nema potrebe praviti novi tip štampača, dovoljno je samo kreirati novi tip *PrinterWithFonts* koji je kao naš „obični štampač" samo što još može da mjenja stilove štampe. Novi tip će naslijediti sve osobine starog, ali će biti u stanju da izvede i druge oblike operacija.

Polimofizam (eng. *polymorphism*) – pošto je *PrinterWithFonts* već ionako *Printer*, nema razloga da ostatak programa „ne vidi" njega kao običan štampač, sve dok mu nisu potrebne nove mogućnosti štampača. Ranije napisani djelovi programa koji koriste tip Printer ne moraju se uopšte pretpostavljati, oni će jednako dobro raditi i sa novim tipom. Karakteristika da se novi tip „odaziva" na pravi način, iako ga je korisnik „pozvao" kao da je stari tip, naziva se polimofizam.

S obzirom na to da je C++ bolja „inačica" C-a, on opet nije „čisto" objektno orjentisan programski jezik. Stvar je u tome što C++ omogućava realizaciju kompleksnog sofvera upotrebom OOP koncepta dosta efikasnije nego što je to bio slučaj sa proceduralnim programiranjem (Paskal je jedan takav programski jezik). Navedene karakteristike OOP su prijeko potrebne ukoliko se pristupa razvoju igara koje koriste sve navedene koncepte ,a znamo kako je C++ industrijski standard kada se govori o toj oblasti informatike.

Kako da...?

Možemo na kraju sumirati kako C++ u kombinaciji sa OOP tehnikom programiranja omogućava razvoj mnogo boljeg, stabilnijeg softvera koji je lako portabilan, multi platformski, lak za debagovanje itd. S druge strane, OOP uvodi drugačiji način razmišljanja programiranja, gdje se na kompleksne cjeline softvera gleda kao na realne objekte što omogućava programerima da rade na jednom objektu, realizaciji jedne funkcije softvera te sklapanje tih objekata u jednu cjelinu čime se dobija gotov softver. Dakle, u OOP-u, razmišlja se o djelovima sistema (objektima) koji nešto rade, a ne o tome kako se nešto radi (algoritmima). Pažnja se prebacuje sa realizacije na međusobne veze između djelova. Nastoji se smanjiti interakcija između softverskih djelova itd. U drugom dijelu ćemo se upoznati detaljnije sa pojmom objekata i klasa, načinom njihove izrade i primjene u C++ programskom jeziku kroz praktičan primjer.

Ki-Pas iz komandne linije

Autor: Marjan Đuran

Uvod

Ako vam se svideo Ki-pas-iks, a bavite se administracijom linux servera, imate samo poneki lični server, ili jednostavno ne volite grafičko okruženje, za vas postoji odlično rešenje - kpcli (eng. *KeePass Command Line Interface*), dakle Ki-pas-iks iz komandne linije. Može se naći na svim repozitorijumima na kojima je i Ki-pas-iks, kao i na zvaničnoj stranici.

Instalacija bi izgledala ovako:

Na fedora (baziranim) sistemima:

```
sudo dnf install kpcli
```

Na centos sistemima:

```
sudo yum install kpcli
```

Na debijan orijentisanim sistemima:

```
sudo apt-get install kpcli
```

Na arč sistemima:

```
pacman -S kpcli
```

Što se kriptografskog dela tiče, model je isti, te je zadržana i kompatibilnost sa Ki-pas-iks bazom, dakle - samo okruženje je promenjeno, a predstavljanje okruženja sledi.

Upotreba

Nakon instalacije u komandnoj liniji ukucajte kpcli. Odziv će biti ovakav:

```
 KeePass CLI (kpcli) v3.0 is ready for operation.
 Type  'help'  for a description of available commands.
Type  'help  <command>'  for details on individual commands.
```

Ovom komandom smo „ušli u program", komandom *help* možemo dobiti listu ostalih komandi uz pomoć kojih možemo da manipulišemo našom bazom kredencijala. Ono što je bitno imati u vidu u samom početku, a što se može videti i kroz Ki-pas-iks i grafičko okruženje prikazano u prethodnom tekstu, jeste analogija Ki-pas aplikacija i stabla, odnosno rasporeda direktorijuma u samom sistemu. Kao što u Linuks operativnim sistemima sve kreće od korenog, odnosno *root* direktorijuma, tako je i u Ki-pas aplikacijama u samom korenu - koren, odnosno *root* direktorijum. Sve grupe i podgrupe možemo posmatrati kao direktorijume odnosno poddirektorijume, a same zapise naših kredencijala možemo posmatrati kao datoteke. Ovo je dosta bitno prihvatiti radi lakšeg snalaženja u kpcli programu.

Zanimljivo je to da su neke komande identične onima iz samog Linuks sistema, kao što su:

- ls - izlistavanje sadržaja u trenutnom direktorijumu,

- pwd - prikaz u kojem se direktorijumu trenutno nalazimo,

- cd - (*change directory*) - promena direktorijuma,

- cl - promena direktorijuma i izlistavanje novootvorenog direktorijuma, zanimljiva komanda nastala spajanjem komandi *cd* i *ls*, dakle kao kada bismo napisali u jednoj liniji *cd; ls*

- mkdir - (*make directory*) - pravljenje novog direktorijuma, odnosno grupe,

- rmdir - (*remove (empty) directory*) - brisanje direktorijuma, odnosno grupe,

- rm - (*remove*) - brisanje jedne datoteke, odnosno zapisa kredencijala iz baze.

No, da pređemo na praktični deo kroz koji se po (nepisanom) pravilu uvek bolje

nauči i upozna novo okruženje ili aplikacija. Postavićemo dva slučaja, pa ćemo ih i kombinovati, doslovno.

Prvi slučaj

Prvi slučaj može biti da smo potpuno novi korisnik i da nemamo nikakvu bazu, ali želimo da je oformimo i u njoj čuvamo svoje kredencijale. Pri tome ne volimo ili nemamo mogućnost grafičkog korisničkog okruženja. Na već opisani način smo na svom operativnom sistemu instalirali kpcli, po istom receptu i pokrenuli aplikaciju. Šta sledi? Pravljenje nove baze. Zapravo, kada ukucamo kpcli, mi smo otvorili novu bazu, ako ukucamo komandu *ls*, odziv će biti:

```
kpcli:/>  ls
=== Groups ===
eMail/
Internet/
```

To su standardne dve grupe koje se nalaze u praznim bazama, koje su tu da nas usmere kako da razmišljamo, odnosno razdvajamo zapise po grupama i pravimo nove. Ukoliko nam se ove grupe ne sviđaju, možemo ih obrisati komandama:

```
rmdir eMail
rmdir Internet
```

Mala digresija na razliku između komande *rmdir* u samom Linuks operativnom sistemu i u programu kpcli - u operativnom sistemu ova komanda briše prazne direktorijume i može obrisati više direktorijuma jednom komandom. U programu kpcli ova komanda briše direktorijum odnosno grupu kredencijala bez obzira da li je prazan ili pun, i može prihvatiti maksimalno jedan argument, odnosno samo jedna grupa se može obrisati u jednoj komandi i zato smo u primeru za brisanje dve grupe komandu pokrenuli dva puta.

Napravićemo novu grupu, odnosno direktorijum, zatim ući u njega i napraviti novi zapis za svoj (izmišljeni) *Prvi nalog*:

```
mkdir Libre!
cd Libre!
new Prvi
```

Odziv koji dobijam izgleda ovako:

```
kpcli:/Libre!>  new Prvi
Adding new entry to  "/Libre!" itle: Prvi
Username: mojeKorisnickoIme
Password:
```

Prilikom izbora šifre, imamo tri mogućnosti. Ako već imamo šifru, možemo samo da je upišemo, pritisnemo *Enter* i idemo dalje, međutim ako ovom prilikom želimo i da generišemo šifru (u čemu su Ki-pas aplikacije prilično dobre), to možemo uraditi na dva načina - odabirom tastera **g** ili **w** za automatsko generisanje šifre, ili opcija **i** koja označava interaktivno generisanje šifre. Razlika između dve opcije automatskog generisanja je što ćemo opcijom **g** dobiti nasumično generisan niza karaktera, dok ćemo opcijom **w** dobiti nasumično generisane reči odvojene tačkom. U slučaju odabira opcije **i** dobićemo odziv koji izgleda ovako:

```
INTERACTIVE PASSWORD GENERATION:
 <n>o/<ret>  Do not accept current password, generate another one.
 <y>es       Accept current password.

 <t>oggle    Toggle between random characters and word-based mode.
 <c>ancel    Abort interactive password generation mode.
And in random characters mode:
  +/-       Increase/decrease password length. May be prefixed with a count.
  [n]=      Set password length to [n] chars. If not given, resets to 20 chars.
70eWY95zbybRw_tNf9fE  +/-/= n/y/t/c
```

Ovom nam govori sledeće: opcijom **n** (*no*) odbijamo da prihvatimo trenutno generisanu šifru koja nam je prikazana, i u tom slučaju se šifra ponovo generiše i tako svaki put dok ne prihvatimo opcijom **y** (*yes*) ili ne odbijemo opcijom **c** (*cancel*). Imamo i dodatne opcije, a to su **t** (*toggle*) čime generišemo šifru koja je kombinacija gore pomenutih opcija **g** i **w**, pritiskom tastera **+** uvećavamo broj karaktera koji će se koristiti za šifru, odnosno tasterom **-** umanjujemo broj karaktera, dok možemo i eksplicitno navesti broj karaktera koji želimo da koristimo unosom celog broja. Nakon generisanja šifre, sledi unos URL-a (*Uniform Resource Location*), odnosno internet adrese lokacije za koju unosimo ili generišemo kredencijale. U sledećem polju će biti mesto za komentar.

Komentar je polje za više redova, završavamo ga tačkom **.** i tasterom **Enter**. Ukoliko ne želite da pišete bilo kakav komentar, samo unesite tačku **.**. Ovime smo ujedno i završili unos naših prvih kredencijala. Na redu je čuvanje unosa. Preporučuje se čuvanje nakon svakog novog unosa, jer ne želite da unesete i generišete preko deset kredencijala. a onda nestane struje, izgubite konekciju sa udaljenim serverom ili jednostavno zaboravite da sačuvate jer ste se zaneli u poslu ili vas je nešto omelo i odvuklo vam pažnju. Prvi put kada čuvate novu bazu, potrebno je da to uradite komandom **saveas**, minimalan unos ove komande je uz parametar koji će biti ime baze, a maksimalan sa dva parametra gde će drugi biti ključ ukoliko na taj način želite da zaštitite svoju bazu. Mi ćemo uraditi samo jednim parametrom, unosom imena.

```
saveas Libre!kpcli
```

Zatražiće nam nakon toga da unesemo *master password*, odnosno glavnu šifru koja će štititi sve ostale kredencijale, i naravno da je verifikujemo, odnosno unesemo još jednom kako bismo otklonili mogućnost da smo neko slovo promašili i time zaključali svoje kredencijale, možda i trajno. Potrudite se da to bude nešto samo vama smisleno i logično, sa dovoljnim brojem što različitijih karaktera, ne dozvolite da izložite opasnosti sve svoje kredencijale zbog manjka maštovitosti i kreativnosti u datom trenutku. Više o šiframa možete pročitati u Libre! tekstu o šiframa.

Baza je sačuvana u direktorijumu u kojem ste se nalazili kada se ukucali komandu kpcli i time pokrenuli program. Naravno, možete naknadno da je premestite gde god želite, ili joj promenite ime. Ukoliko bismo sada uneli još neke kredencijale u *istu* bazu, nakon unosa komentara, što je poslednji korak, dobili bismo obaveštenje i pitanje:

```
Database was modified. Do you want to save it now? [y/N]:
```

Baza je izmenjena. Da li želite da sačuvate izmene sada? Odgovaramo sa **y** ukoliko želimo, ili **n** ukoliko ne želimo. Zanimljivo je da možemo odgovoriti i sa **n**, zapis će biti prisutan nakon završetka unosa kredencijala, to možemo proveriti komandom **ls**. Međutim, sada je potrebno da unesemo i komandu **save**. Dobijamo odziv:

```
Saved to Libre!kpcli
```

Kako da...?

Šta se dešava ako smo imali novi unos, nismo želeli odmah da sačuvamo unos, ali nismo ni komandu *save* upotrebili? Prilikom izlaska iz programa, što se radi komandama **close**, **quit** ili **exit**, dobićemo poslednje upozorenje koje izgleda ovako:

```
WARNING: The database has changed and was not saved.
Really close it? [y/N]
```

Ovo nam je poslednja šansa da sačuvamo novounete izmene. Opcijom „y" potvrđujemo da želimo da sačuvamo, dok opcijom „N" koja je velikim slovom, što nam govori da je ona podrazumevana, dakle možemo uneti malo ili veliko slovo N ili samo pritisnuti Enter), naše izmene u bazi neće biti sačuvane. Komandom exit izlazimo iz programa.

Numerička obrada i simulacije (8. deo)

Autor: Stefan Nožinić

Šta dalje?

U ovom, poslednjem članku ovog serijala Vam dajemo primere i ideje kako biste mogli da nastavite istraživanje i učenje ako vas ova oblast zanima.

Mašinsko učenje

Mašinsko učenje podrazumeva statističku obradu podataka koja se kasnije primenjuje na prepoznavanje obrazaca od strane računara. Ovo podrazumeva da računar može da prepoznaje neke njemu naizgled netrivijalne stvari poput lica, glasa, objekata na slici pa sve do predviđanja temperature i stanja na berzama. Postoje razne metode mašinskog učenja, od skrivenih Markovljijevih modela pa sve do neuronskih mreža. Nedavno je posebno aktuelna oblast dubinsko učenje.

Vizuelizacija podataka

Danas su podaci svuda oko nas i svima su dostupni. Problem je dobro organizovati podatke i zaključiti nešto smisleno od njih. Vizuelizacija se bavi upravo tom tematikom. Ovo je jedna od oblasti gde je potrebno imati malo i umetničkog dara kako biste određenu pojavu što bolje prikazali.

Simulacije fizičkih sistema

O ovome smo već dosta pisali. Simulacije fizičkih sistema imaju veliku ulogu u raznim drugim oblastima. Pre svega u elektronici, mehanici, meteorologiji, ali i u industriji video igara gde je realan doživljaj veoma važan kako bi se igrač osećao

što zadovoljnije.

Šta preporučujemo za čitanje?

U ostatku teksta vam navodimo literaturu za koju mislimo da dobro objašnjavaju ove oblasti:

C. M. Bishop - Pattern recognition and machine learning - knjiga objašnjava različite metode mašinskog učenja. Veoma je bogata sadržajem i daje dobre primere gde se svaki metod primenjuje.

H. Beigi - Fundamentals of speaker recognition - objašnjava metode prepoznavanja govora i govornika.

L. Deng D. Yu - Deep learning methods and applications - uvodi čitaoce u oblast dubinskog učenja koje je danas dosta popularno.

M. Sonka V. Hlavac R. Boyle - Image processing analysis and machine vision - prikazuje razne metode obrade slike i računarske vizije.

S. Z. Li A. K. Jain - Handbook of face recognition - priručnik koji sadrži metode za prepoznavanje i detekciju lica.

D. Bindel J. Goodman - Principles of scientific computing - daje prikaz numeričke obrade sa detaljnim objašnjenima različitih metoda i analizama grešaka.

S. R. Buss - 3D computer graphics - a mathematical introduction with OpenGL - jedan deo dosta dobro objašnjava matematički deo linearnih transformacija i njihovu primenu u izradi igara.

Naredbe u GNU-Linuksu (3. dio)

Autor: Adrijan Đurin

Kako smo većinu vremena u prošlom članku proveli stvarajući, u ovom članku ćemo se upoznati sa dvije naredbe koje služe za brisanje datoteka i direktorija. Brisanje je važno jer ne želimo da nam određeni direktoriji postanu zatrpani kojekavim datotekama i direktorijima koji nam više ne koriste. Objasnit ćemo i naredbe za kopiranje, preimenovanje i premještanje datoteka. Jedna opaska pri korištenju naredbi za brisanje: **datoteke i direktoriji bivaju trajno obrisani**. Dakle, ako se određene datoteke i direktoriji obrišu korištenjem sljedećih naredbi, oni su trajno obrisani. Nisu, kao na vindouzu, poslani u kantu za smeće iz koje se mogu povratiti. Stoga, oprez.

Prva u nizu u naredbi je **rmdir**. Ako upišemo:

```
man rmdir
```

vidjet ćemo da u opisu naredbe piše da ova naredba služi za brisanje praznih direktorija. Ukoliko direktorij nije prazan, javit će vam se sljedeća greška:

```
rmdir: failed to remove  'pizzerija/': Directory not empty
```

Ako ste pokušali zadnju naredbu iz prošlog članka, primjetili ste da su kreirana dva prazna direktorija, *Moje/* i*pjesme/* . Kako kreirati takav direktorij bit će objašnjeno kasnije. Za sada, isprobajmo ovu naredbu **rmdir**.

```
rmdir: failed to remove  'pizzerija/': Directory not empty
```

Kao što vidite, naredbi je moguće predati i više od jednog argumenta. U ovom

slučaju, dva direktorija. To sve piše u priručniku za **rmdir** naredbu. Za brisanje pojedinačnih datoteka koristimo **rm** naredbu. U opisu naredbe stoji kako služi za brisanje datoteka i direktorija. Za brisanje neke datoteke, sintaksa je ista kao i kod **rmdir**. Kreirajte praznu datoteku i onda je obrišite koristeći **rm** naredbu.

```
rm lista_cvijeca.txt
```

Naredba rm nudi puno više mogućnosti, pa se stoga ona i češće koristi. O tome kako obrisati direktorije u kojima se nešto nalazi, više u idućem članku.

Sljedeća u nizu naredbi je kopiranje datoteka i direktorija. Naredba **cp** služi za to. Sintaksa cp naredbe je *cp <što želimo kopirati> <gdje želimo kopirati>*. Dakle, poslije **cp** naredbe uvijek navodimo dva argumenta. Recimo da želimo kopirati naš popis stvari za kupovinu u *Documents/* direktorij. Naredba je sljedeća:

```
cp veoma_lijep_popis.txt Documents/
```

Izlistavanjem sadržaja direktorija *Documents/* možete se uvjeriti da je datoteka kopirana. Na isti način se mogu kopirati i prazni direktoriji. Veoma je očito da se ovim načinom stvaraju kopije. Da ne bismo poslije svake **cp** naredbe brisali kopiranu datoteku, postoji jedna naredba koja samo premješta datoteku ili direktorij na odredište koje specificirate. To je naredba **mv**. Sintaksa naredbe je ista kao i kod **cp** naredbe. Poslije mv naredbe slijede dva argumenta: putanja do datoteke ili direktorija koje želimo premjestiti i onda odredišna putanja. Primjer:

```
mv Documents/veoma_lijep_popis.txt Pictures/
```

Ovom naredbom ste ranije kopiranu datoteku prebacili u drugi direktorij. Da se vratimo na problem s početka članka, a to je kreiranje datoteka i direktorija s razmakom u imenu. To se može na nekoliko načina. Jedan od njih je da ime datoteke ili direktorija stavimo između navodnika. Primjer:

```
touch  "iduci put pametnije".txt
```

```
mkdir  "moje putovanje Belgijom"
```

Naredbom za izlistavanje možete primjetiti kako se takva datoteka prikazuje s navodnicima. To sugerira da se radi o samo o jednoj datoteci koja ima praznine u imenu. Isti princip vrijedi i za direktorije.

Za kraj, opet nekoliko naredbi da se malo zabavite i razmislite. Što se događa kad unesete sljedeće naredbe:

```
cp veoma_lijep_popis.txt
veoma_lijep_popis_2.txt
```

```
mv veoma_lijep_popis.txt
veoma_lijep_popis_3.txt
```

```
clear
```

Pregled popularnosti Gnu-Linuks i BSD distribucija u poslednjih šest meseci

Distrowatch

1	Mint	2870<
2	Debian	2196>
3	Manjaro	1805<
4	Ubuntu	1264<
5	Antergos	1186<
6	openSUSE	989<
7	Deepin	960<
8	Fedora	949<
9	TrueOS	891=
10	elementary	830<
11	Zorin	818<
12	Solus	782<
13	MX Linux	681=
14	Arch	654<
15	PCLinuxOS	650>
16	ReactOS	631<
17	CentOS	610<
18	Peppermint	554<
19	Mageia	551<
20	Kali	495<
21	Devuan	458<
22	AntiX	444>
23	Lite	429<
24	KDE neon	420<
25	Alpine	412>

Pad <
Porast >
Isti rejting =
(Korišćeni podaci sa Distrovoča)

FAN (2.deo)

Nagios

Autor: Stefan Biševac

U prvom članku uveli smo vas u pojam i definiciju FAN-a. Na slici 1 prikazane su komponente datog sistema. Kao što smo i obećali, u narednim tekstovima pisaćemo o pojedinačnim komponentama ovog sistema, dati uvod u osnovne pojmove i načelne konfiguracije svake komponente ponaosob. Prva i najvažnija je Nagios pa će u narednom delu teksta biti iznete najvažnije stavke ove komponente. Kolege koje već koriste Nagios jezgro (eng. *Nagios Core*) ovaj deo teksta mogu da preskoče ili da se pak podsete nekih stvari koje su im postale rutine u radu sa Nagiosom.

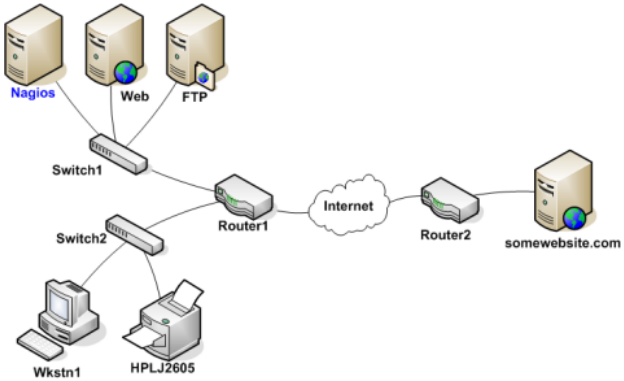
S obzirom da se FAN može preuzeti sa zvanične stranice FAN organizacije kao nezavistan sistem koji se instalira na CentOS operativnom sistemu, smatramo da nije potrebno prikazivati instalacije svake komponente. U samom paketu FAN-a dolaze unapred instalirane sve tri. Ukoliko koristite već podešen Nagios u

samostalnoj verziji Nagios jezgra, naš savet je da konfiguracione datoteke prebacite na Nagios koji dolazi u paketu FAN i izvršite minimalna, neophodna podešavanja. Kolege koje do sada nisu koristile Nagios nemaju o čemu da brinu jer u narednom delu teksta sledi uvod u Nagios.

Nagios

Nagios je centralna komponenta FAN sistema i kao takva praktično predstavlja njegovu centralnu jedinicu, tj. kompletna obrada podataka i odlučivanje vrši se u Nagios jezgru. Da bismo uopšte nastavili o detaljima, krenućemo u obrnutom smeru, od vrha ka dnu, pokazaćemo gde je mesto FAN-a u mrežnoj infrastrukturi do načina na koji Nagios funkcioniše.

Slika 2, prikazuje mesto i ulogu Nagios monitoring sistema u mreži. Najčešće se postavlja u kontrolisanoj zoni mreže i bliže uređajima na kojima se izvršavaju servisi od velikoj značaja.

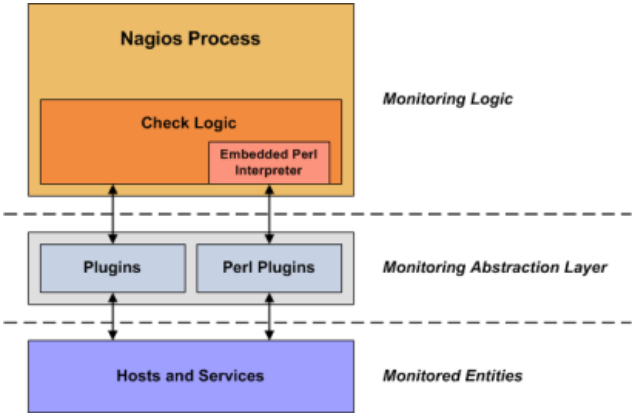


Na slici 2, prikazana je logička šema mrežne infrastrukture. Dakle, Nagios može da prati stanje fizičkih uređaja kao i servisa koji se na njima izvršavaju. Da bi Nagios uopšte funkcionisao kao monitoring sistem, on mora imati pristup svakom uređaju nad kojim se vrši nadzor ili da postoji posrednik koji ima vezu sa željenim uređajem i koji će za njega izvršavati proveru, a zatim vraćati informaciju u unapred definisanom obliku. Ovde se dolazi do prvog važnog pitanja, a ono glasi: „Ukoliko želimo da Nagios vrši direktne provere nad uređajima koji se nalaze u zasebnim virtuelnim mrežama (VLAN) koje pak nisu

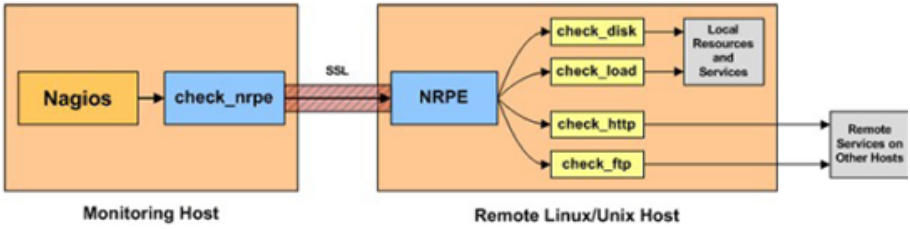
rutabilne, kako da prevaziđemo ovaj problem s obzirom na to da je Nagios dodeljen jednoj od virtuelnih mreža?". Odgovor se krije u samoj mrežnoj infrastrukturi. Ukoliko imate fizičke servere, tada će vam posao biti težak jer će broj virtuelnih mreža određivati broj fizičkih kartica ili ćete pak morati da između određenih virtuelnih mreža uspostavite konektivnost. Ukoliko imate virtuelizovanu infrastrukturu, tada se posao svodi na dodavanje vrituelne mrežne kartice Nagios virtuelnoj mašini. Vodite računa da Nagios server može da ima mrežni prolaz (eng. *gateway*) samo na jednoj mrežnoj kartici. Shodno prethodnom tekstu utvrdili smo gde je načelna pozicija Nagios servera. Ono što sada treba objasniti jeste kako Nagios funkcioniše, tj. na koji način vrši nadzor uređaja i servisa. Postoji razlika kada su u pitanju sistemi nad kojima se vrši nadzor pa tako dolazimo do užeg definisanja infrastrukturnih komponenti. To su serveri na kojima se izvršavaju Vindouz i Linuks operativni sistemi, a sa druge strane to su mrežni uređaji koji moraju posedovati SNMP protokol, inače Nagios neće biti u stanju da monitoriše i upravlja datim mrežnim uređajima.

Monitoring mašina

Nagiosov princip rada dat je na slici 3. U prvom sloju nalazi se Nagios jezgro, tj. logika monitoringa. Iz ovog sloja se upućuju zahtevi za proverom uređaja i servisa. Da bi Nagios bio u stanju da vrši provere nad nezavisnim mrežnim subjektima, on mora posedovati pomoćne programe (plugins) kojima izvršava specifične naredbe. Pomoćni programi se mogu nalaziti na Nagios strani ili na strani udaljenih servera u zavisnosti od informacija koje su nam potrebne.

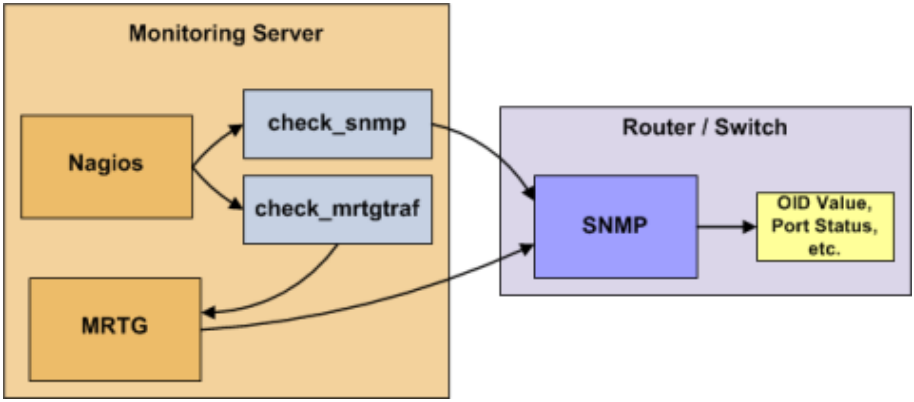


Nagios Remote Plugin Executor predstavlja agent koji je namenjen za izvršavanje pomoćniih programa na Linuks operativnom sistemu. Kod Linuks mašina postoji i provera koristeći SSH, međutim, sama upotreba ovog protokola zahteva i snažnije resurse Nagiosa pogotovo kada je reč o velikom broju provera putem ovog protokola. Sistem rada sastoji se iz nekoliko faza: Nagios putem pomoćnog programa **check_ nrpe** izdaje set naredbi udaljenom uređaju na kojem se nalazi *NRPE*, prihvaćen set komandi se izvršava i dobijene informacije se vraćaju nazad do Nagiosa. Ova jednostavna arhitektura ostavlja velike mogućnosti sistem administratorima da sami napišu pomoćne programe, a da se njihovo izvršenje svede na sistem rada *NRPE*. Sličan način rada definisan je i za Vindous servere koji koriste **check_nt** pomoćni program i *NSClient++* agent.



NRPE i *NSClient++* imaju istu ulogu u funkciji Nagios nadzora. Više detalja o ovim agentima moguće je pronaći na zvaničnim veb stranicama ovih slobodnih softverskih proizvoda. Ostalo je da definišemo i provere koje se vrše nad mrežnim uređajima. Za ovaj tip provera Nagios koristi pomoćne programe **check_snmp** i **check_mrtgtraf**. Ulogu agenta u ovom slučaju igra SNMP protokol na strani mrežnog uređaja. Nagios izda zahtev za specifičnom proverom po određenom objektu u MIB bazi. SNMP vrati informaciju do Nagios jezgra koja biva prikazana u veb okruženju. Konkretno podešavanje biće prikazano u narednim tekstovima sa detaljima i specifičnostima koje nosi ovakav tip provere. Slika 5 grafički ilustruje princip rada Nagiosa koji vrši proveru putem SNMP protokola.

Slobodni profesionalac



Konfiguracija Nagiosa vrši se kroz njegove konfiguracione datoteke. Ukoliko ne vladate dobro Linuks okruženjem, tada u pomoć stiže *Centreon* komponenta FAN sistema koja će biti objašnjena u narednom članku. Za one napredne i radoznale, preporučujemo da ipak prouče Nagiosove konfiguracione datoteke kako bi imali potpuni uvid u rad Nagiosa jer je *Centreon* komponenta koja zamenjuje konzolu grafičkim pristupom datotekama. Time se smanjuje mogućnost greške prilikom konfiguracije koju korisnik unosi pisanjem po datotekama Nagiosa.

Nagios®

Jezička laboratorija u Raketu (1. deo)

Autor: Luka Hadži-Đokić

Pravljenje jezika - zašto?

Programskih jezika je mnogo. Može se reći da je zbog toga pravljenje novih jezika beskorisno. Međutim, malo koji od njih je zamišljen kao programski jezik opšte upotrebe, nego prije kao alatka za rešavanje vrlo specijalizovanih problema - kao domen-specifični programski jezik. Oni su našli upotrebu u raznim poljima - od digitalnog pisanja muzike, preko veb-dizajna, do interakcije sa robotima.

Da bismo razumeli kako kompjuterski jezici rade, u ovom serijalu ćemo pokušati da napravimo jedan mali programski jezik.

Šta je jezik?

Jezik je, u formalnom smislu, skup simbola (alfabet) i skup pravila za pravljenje reči od tih simbola (gramatika). Kao što vidite, ovakva definicija se ne ograničava samo na programske jezike. Primer gramatike jednog ograničenog dela srpskog jezika bi bio sledeći:

- Rečenica je subjekat, praćen predikatom i objektom i završava se znakom

interpunkcije;

- Subjekat je ili “Milan” ili “Dragan”;

- Predikat je jedan od “jede”, “priča”, “pravi” i “nosi”;

- Objekat je jedan od “sarmu”, “priču”, “programski jezik” i “ranac”;

- Znak interpunkcije je “.” ili “!”.

U ovako definisanom jeziku, “Milan pravi sarmu.” je validna rečenica. Međutim, možemo napraviti i neke besmislene rečenice, kao što je “Dragan jede programski jezik!”. Ovo može dovesti do zabune. Međutim, treba imati na umu da gramatika određuje samo sintaksu jezika. Kada pričamo o smislu rečenice, dotičemo se semantike, kojom dodeljujemo značenje znakovima, rečima ili celim rečenicama.

Gramatiku programskih jezika predstavljamo na sličan način, u takozvanoj Bakus-Naurovoj formi (o kojoj će biti više priče u sledećim delovima), dok je semantika ono čime se određuje kako se program izvršava.

Kako će raditi prevodilac

Kao što ste možda do sada čuli, prevodilac ili kompajler (eng. *compiler*) je program koji izvorni kod u nekom programskom jeziku prevodi u mašinski jezik računara. Interpretatori, slično tome, izvorni kod direktno izvršavaju. Naš jezik ćemo prevoditi u Raket (eng. *Racket*), funkcionalni programski jezik iz LISP familije.

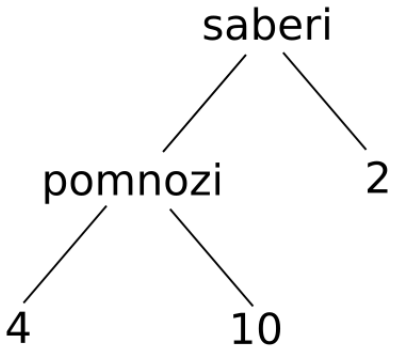
Ovaj proces biće podeljen u 2 dela (koji će biti dalje podeljeni u kasnijim tekstovima):

1. Čitač (eng. *reader*) će transformisati kod našeg jezika u S-izraze (eng. *C-expressions*), koji su glavni oblik izraza (koda koji može da se izvrši) u Raketu. Primer jednog S-izraza je:

```
(saberi (pomnozi 4 10) 2)
```

Takođe su vrlo bliski apstraktnim sintaksnim stablima, koja služe za

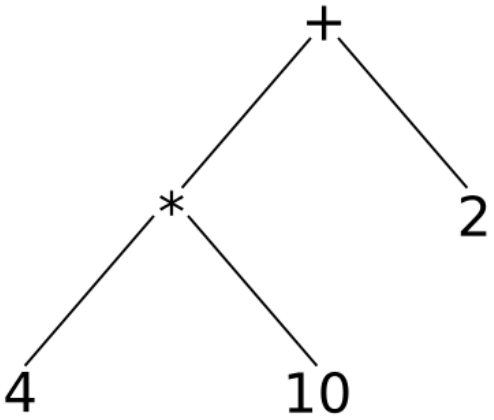
predstavljanje koda u opštem slučaju, pa se može predstaviti i ovako:



2. Ekspander (eng. *expander*) pretvara S-izraz u validne Raket izraze, kako bi mogli da ih izvršimo. Tako će prethodni S-izraz biti transformisan u

```
(+ (* 4 10) 2)
```

ili u obliku stabla:



koje se može izračunati, i rezultat je 42.

Mali uvod u Raket

Kako bi pratili naredne tekstove, potrebno je da instalirate Raket i upoznate se sa njim. Instalaciju možete naći na sledećem linku, ili u u softverskim centrima vaše distribucije. Ova instalacija sadrži

- raket (eng. *racket*) - prevodilac,

- rako (eng. *raco*), menadžer paketa za raket, kao i

- DrRaket (eng. *DrRacket*), interaktivno razvojno okruženje (eng. *interactive development environment* ili *IDE*).

Okruženje DrRaket-a podeljeno je na 2 dela: tekstualni editor (gore) i takozvani REPL, petlju čitanja, evaluacije i ispisivanja (eng. *Read Eval Print Loop*), interaktivni Raket interpretator (dole). Na početku Raket programa otvorenog u tekstualnom editoru stoji linija

```
#lang racket
```

Ona omogućava programeru da odredi jezik kojim piše, tj. da odredi čitač i ekspander koje će raket prevodilac koristiti pri prevođenju programa. Za početak, možete se igrati u REPL-u:

```
> (+ 1 1) ; komentari pocinju sa ";", izrazi su u zagradama
2
> '(+ 1 1) ; izrazi su u zagradama, za liste stavljamo ' ispred
'(+ 1 1)
> (define dva 2)
> dva
2
> (let ([dva "Zdravo svete!"]) ; dva ce u okviru
  (printf dva))              ; let izraza biti ovaj string
Zdravo svete!
> dva
2
> (lambda (x) (+ x 40)) ; lambda izrazi predstavljaju funkcije
#<procedure>
> (define funkcija1 ; definisati kao i promenljive
```

```
    (lambda (x)
      (+ x 40)))
  >  (funkcija1 2)
 42
  >  (define (funkcija2 x) (+ x 40)) ; ekvivalentno gornjoj definiciji
  >  (funkcija2 2)
 42
  >  (map funkcija2  '(1 2 3 4)) ; liste su bitne, pa imaju dosta korisnih funkcija
  '(41 42 43 44)
  >  (filter even?  '(1 2 3 4))
  '(2 4)
  >  (if #t        ; #t je true ili tacno, #f je false ili netacno
       "tacno"     ; sve sto nije #f, smatra se tacnim u kondicionalima
       "netacno")
  >  (cond [(>  2 dva) (error  "netacno!")]
        [(<  2 dva) (error  "opet netacno!")]
        [else  "tacno!"]) ; kao if else u drugim programskim jezicima
  "tacno!"
  >  (struct boja (crvena zelena plava)) ; strukture
  >  (define zuta
     (boja 0 255 255))
  >  (boja-plava zuta)
255
```

U narednim tekstovima ćemo, dok pravimo naš jezik, naići na neke zanimljivosti Raketa, ali i funkcionalnih programskih jezika uopšte. Ali ako ne možete da čekate, kratak uvod možete naći ovde.

Kripto-ratovi: Nekada i sada (3. deo)

Autor: Petar Simović

Privategriti

U novije vreme, pojavio se projekat koji bi navodno okončao večne kripto-ratove pod nazivom **Privategriti** (eng. Privategrity). Privategriti bi trebalo da okonča kriptoratove tako što će se implementirati najbolja i najsigurnaija kriptografija u sistem koji čine deset glavnih računara. Poruke i ostali saobraćaj se rutira kroz devet računa, dok je deseti kontrolni, i pruža veću anonimnost i performanse nego postojeće mreže poput Tor-a ili I2P mreže. Svaki od devet računara raspoređen je u drugu državu koja ga kontroliše, što poboljšava anonimnost. Glavna caka je u tome što kada se administratori svih devet računara dogovore, pod pritiskom vlada i zakona, mogu dešifrovati sve poruke i razotkriti ko šta radi. Ovim se sigurnost sistema prelazi iz domena nauke i matematike u ruke birokratiji i politci država koje kontrolišu računare. Time se samo trenutni problem gde jedna kompanija vlasničkog sistema kontroliše privatnost, raspoređuje na devet demokratskih država. Ideja vodilja Privategriti projekta je da se aktivnostima poput terorizma koje su međunarodno okarakterisane kao pretnja može ući u trag u saradnji devet država, dok će privatnost većine građana ostati netaknuta. Ideja zaista zvuči utopijski i verovatno ima potencijala da okonča kripto-ratove, ali zakonski problemi oko toga koji međunarodni organ bi imao pravo da zatraži dešifrovanje komunikacija i dalje ostaje. Sve što ovaj projekat rešava je stavljanje kao da se radi o privatnim firmama koje jedine znaju manu svog sistema i mogu je zloupotrebiti. Ono što je interesantno predstavlja činjenicu da je idejni tvorac projekta niko drugi nego Dejvid Čaum (eng. David Chaum). Isti čovek iz čijih naučnih radova su nastale mreže poput Tora i Miksmaster za zaštitu privatnosti komunikacija ne samo kroz šifrovanje, nego i zavaravanje digitalnih tragova - metapodataka od velikog brata (adversary).

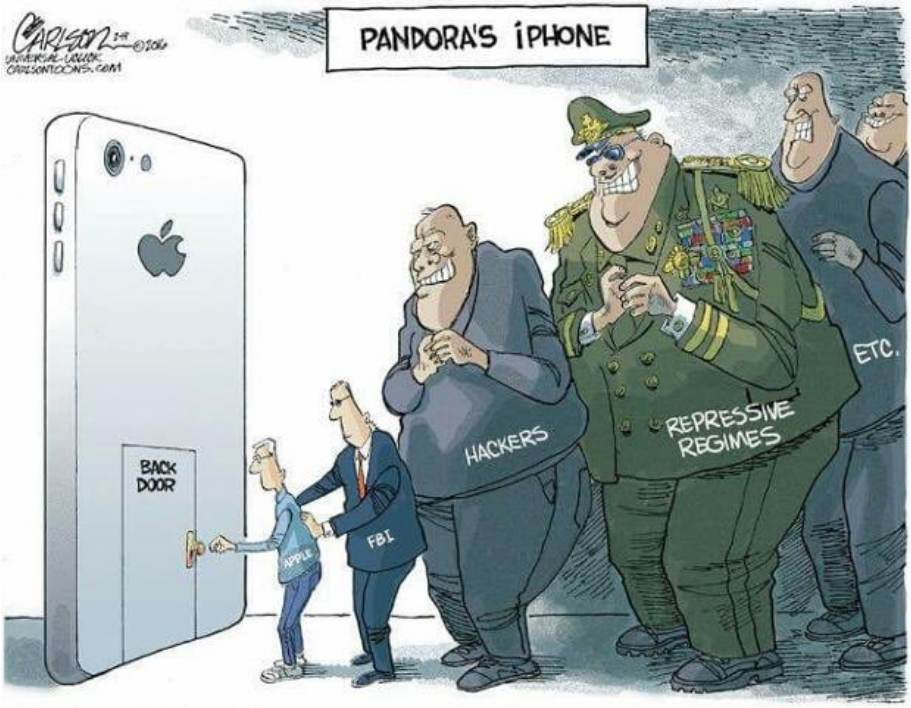
Originalan dokument „Privategriti" projekta možete naći na https://goo.gl/fF3scD, dok za više informacija u medijima možete pročitati na http://goo.gl/s2AvQk i http://goo.gl/dU8oKJ.

Epl protiv Ef-Bi-Aj-a

U najnovije vreme spor između Američkog federalnog istražnog biroa (eng. FBI) i kompanije Epl uzdrmao je i podelio ne samo američku nego i svetsku javnost. U decembru prošle godine desio se teroristički napad u gradu San Bernadio u Kaliforniji gde je jedan od napadača imao šifrovani Eplov telefon. Napadač je poginuo u pucnjavi sa policijom, ali je šifrovani telefon postao glavna tema, jer je Epl od skora uveo sistem šifrovanja svih podataka na telefonu koga ni sam Epl ne može otključati.

Eplov sistem je imao i ograničenje u pokušaju unosa pogrešnih šifara kako bi se obezbedio od napada grubom silom, tj. posle određenog broja unosa neispravnih šifara izbrisali bi se podaci na telefonu. To je naravno zasmetalo istražnim organima jer su im bili potrebni podaci za istragu sa tog telefona, pa su preko suda pokušali da prisile Epl da kroz ažuriranje osperativnog sistema telefona namerno modifikovanom verzijom zaobiđe unos šifre. Time se ispostavilo da je ipak potrebno i poverenje u Epl da je na strani korisnika, što je Epl verovatno i želeo. Pošto Epl nije hteo da to učini i time ugrozi sigurnost svog sistema i privatnost miliona svojih mušterija, pravna bitka se zaoštrila. Na kraju, Epl nije

pomogao Ef-Bi-Aju koji je i bez pomoći Epla otključao telefon. Kako je to Ef-Bi-Aj uradio nije poznato, ali se spekuliše da je hardverski odlemio sporni čip iz telefona i naprednim tehnikama kloniranja čipova obezbedio sebi da isproba veliki broj šifara za dešifrovanje. (http://goo.gl/9WGope, http://goo.gl/8rH2oZ, http://goo.gl/Ah2luq). Interesantan video za ljubitelje američkog humora koji sumira u osamnaest minuta ono o čemu ovde govorimo možete naći na https://goo.gl/v6wmYO.



Vajber i Vatsap

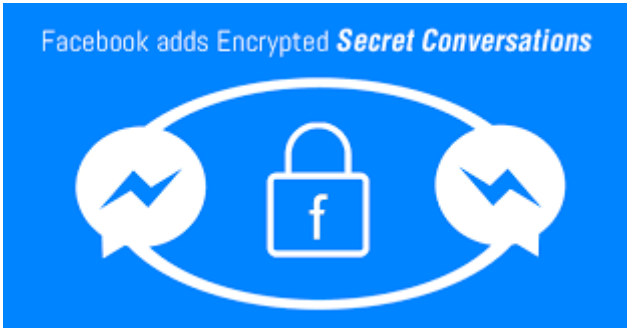
Posle dešifrovanja Eplovog telefona popularni vlasnički servisi Vajber (eng. Viber) i Vatsap (eng. WhatsApp) uveli su E2E (eng. end-to-end) šifrovanje između korisnika njihovih aplikacija. Vatsap implementira isti protokol za šifrovanje kao i popularna aplikacija otvorenog koda Signal neprofitne organizacije Open Visper Sistems (eng. Open Whisper Systems) (http://goo.gl/ZkQVxy, https://goo.gl/gK4BMu). Ovaj potez Vatsap-a prati i konkurentna aplikacija Vajber

koja uvodi istu vrstu zaštite korisničkih podataka za poruke, video i audio-pozive. (https://goo.gl/8XSKCl) E2E zaštita praktično znači da kompanija čiju aplikaciju koristite i preko čijih servera komunikacija ide nema ključeve kojima su šifrovani podaci korisnika dok putuju mrežom. Svi podaci se mogu dešifrovati samo na „drugom kraju" tj. na telefonu sagovornika - korisnika iste aplikacije.

Fejsbuk i Alo

Kako bi zadržali i privukli korisnike kojima je privatnost važna, isti trend šifrovanja prate i fejsbuk i guglov alo (eng. Google Allo). Naime, prvo je objavljeno da gugl radi na svom alo pametnoj aplikaciji za dopisivanje (koja nije otvorenog koda) (https://goo.gl/2w2xcb). Zatim je došla vest o fejsbukovoj aplikaciji pod nazivom *Tajni razgovori* (eng. Secret Conversations) koja omogućava E2E šifrovanje unutar fejsbukovog mesindžera (https://goo.gl/gRkodi). Međutim glavna kvaka je u tome što korisnik nema mnogo načina da se uveri u sigurnost jer su iyvorni kodovi nedostupni, tj.

aplikacije su vlasnički softver. Postoji dosta načina da se zobiđe dobro implementirana kriptografija unutar vlasničkog softvera, i oda vaš tajni ključ vlasniku aplikacije, pa se korišćenje vlasničkog softvera u svrhu razmene poverljivih informacija nikako ne preporučuje.



Blekberi

Kanadska kraljevska policija imala pristup šifrovanim porukama Blekberija (http://goo.gl/AlgygX, https://goo.gl/MTG2V9), a izvršni direktor Blekberija odgovorio da kompanije ne treba da stavljaju svoj ugled iznad viših ciljeva (http://goo.gl/0WIUIV, http://goo.gl/CwJJSI). Još jedna potvrda da je zaštita privatnosti korisnika dobar i unosan posao, u poslednje vreme potvrđuje i slučaj zaplenjivanja servera kompanije Enetkom (eng. Ennetcom) koja je prodavala modifikovane Blekberi telefone i pružala uslugu šifrovane komunikacije ukoliko ste korisnik Enetkom mreže (https://goo.gl/EdpLtb).

Šifrovano ćaskanje na Androidu: Ksaber

Autor: Petar Simović

Aplikacija za ćaskanje ima mnogo, ali pitanje je kako odabrati onu pravu, aplikaciju kojoj možete verovati malo više nego ostalima kada želite da razmenjujete privatne poruke sa drugom osobom. Svakako bi trebalo da tražite one aplikacije koje su otvorenog koda, ili još bolje da aplikacije tražite sa F-Droid. Mi preporučujemo aplikaciju otvorenog koda **Ksaber** (eng. *Xabber*) za privatno šifrovano četovanje na Androidu. Ksaber je XMPP/Jabber klijent, a za šifrovanje preko XMPP-a koristi se **OTR** (eng. *Off-The-Record*) protokol koji je uključen u Ksaber aplikaciju, tako da pored Ksaber-a aplikacije nije potrebno instalirati ništa drugo.

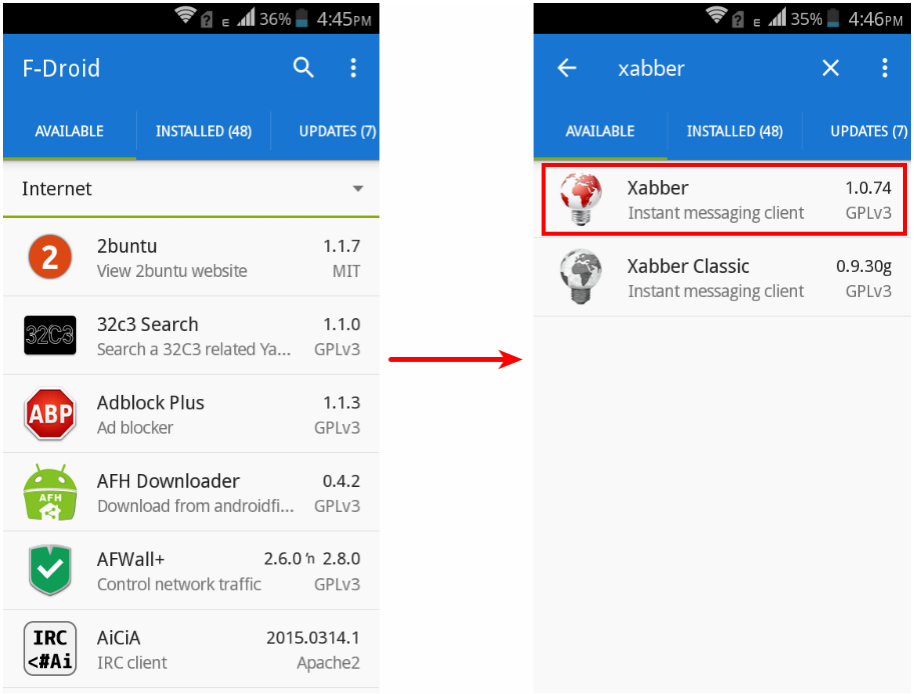
Osim Ksaber-a, postoje i drugi programi koji su takođe otvorenog koda i pomoću kojih možete šifrovano ćaskati pomoću OTR protokola kao što su, na primer, Konverzejšns (eng. *Conversations*), Četsekjur (eng. *ChatSecure*), Zom, Bim (eng. *Beem*) i Žici (eng. *Jitsi*), i koje takođe preporučujemo kao dobre alternative.

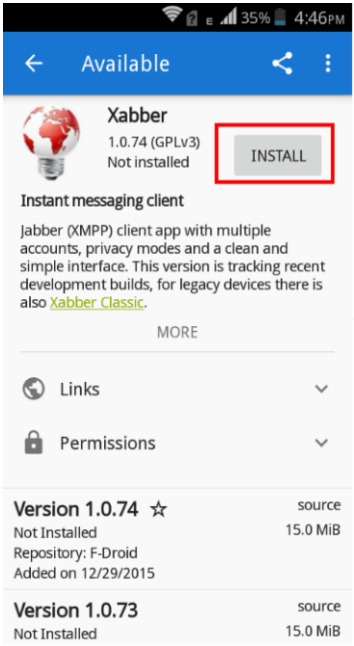
OTR protokol ne podržava grupno šifrovano dopisivanje, kao ni šifrovanu razmenu fajlova, već samo tekstualne poruke. Međutim koristeći OTR možete se sa sagovornikom dogovoriti oko tajne šifre tokom dopisivanja, a zatim drugim programom šifrovati fajl dogovorenom šifrom pre slanja, i tek onda izvršiti slanje.

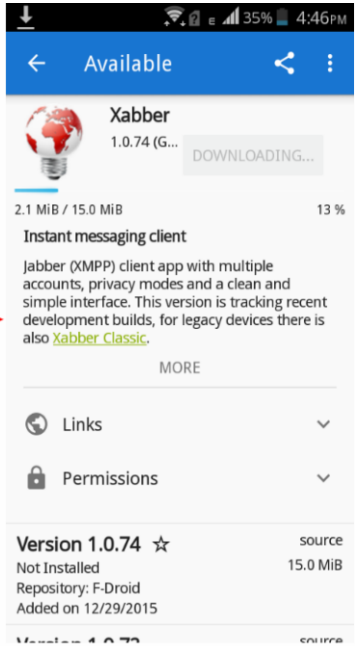
OTR protokol je nezavistan od protokola/servisa koji koristite za komunikaciju pa ćete tako moći da ga koristite za privatnu konverzaciju i preko IRC-a, Gugl Tok-a (eng. *Google Talk*), Jahu Mesindžera (eng. *Yahoo Messenger*) i drugih, dok god i vaš sagovornik koristi isti protokol. Kao što se ne možete dopisivati ako koristite Jahu Mesindžer, a sagovornik IRC, tako i OTR funkcioniše samo ako ga koriste obe strane u komunikaciji.

Instalacija Ksabera

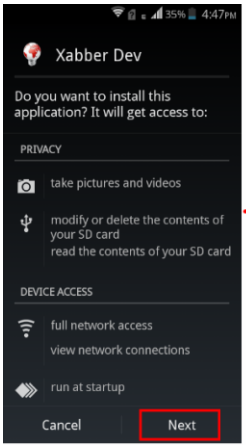
Ksaber možete preuzeti i instalirati sa GuglPlej-a (eng. *Google Play*) ili F-Droid-a, a mi svakako preporučujemo da to uradite sa F-Droid-a. Pronađite Ksaber na F-Droid-u i instalirajte ga.

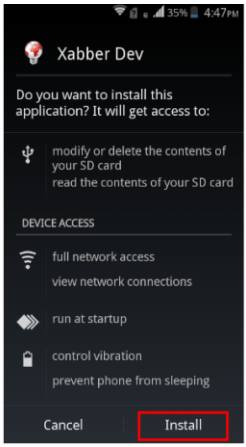


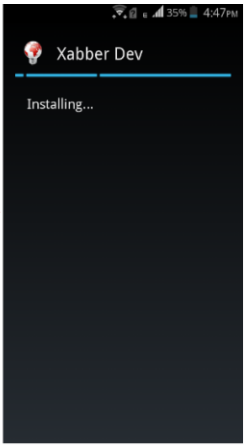


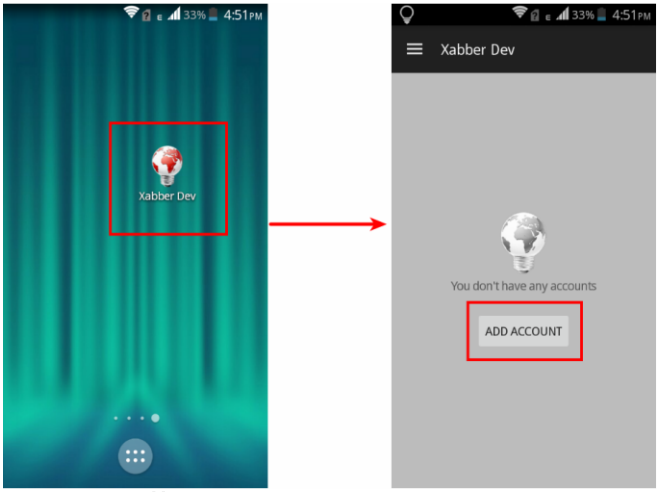


U toku instalacije Ksaber će vas pitati za privilegije koje su mu potrebne da bi normalno funkcionisao i, ako mu iste odobrite, instaliraće se. Kada se Ksaber instalira, pokrenite ga i pritisnite dugme za dodavanje novog naloga.



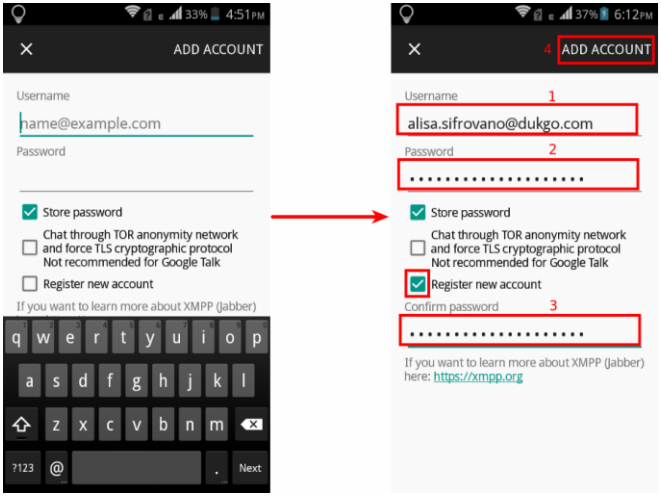






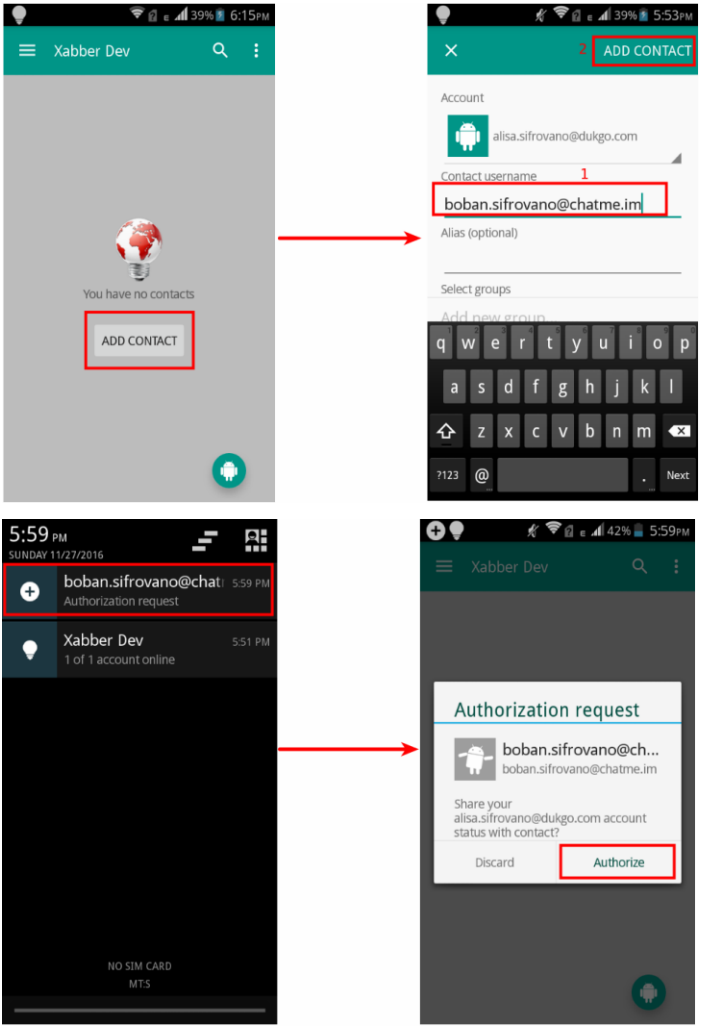
Dodavanje vašeg naloga i kontakta

Dalje je potrebno uneti vaš XMPP nalog ukoliko ga imate, ili možete čekirati kvadratić „Register New Account" kako bi sa unetim podacima registrovali na željenom XMPP serveru vaš novi nalog. Ako ne znate koji XMPP server da koristite, listu javnih XMPP servera možete naći na list.jabber.at. Naš nalog je *alisa.sifrovano@dukgo.com*, a vi unesite svoj.



Dodavanje kontakta

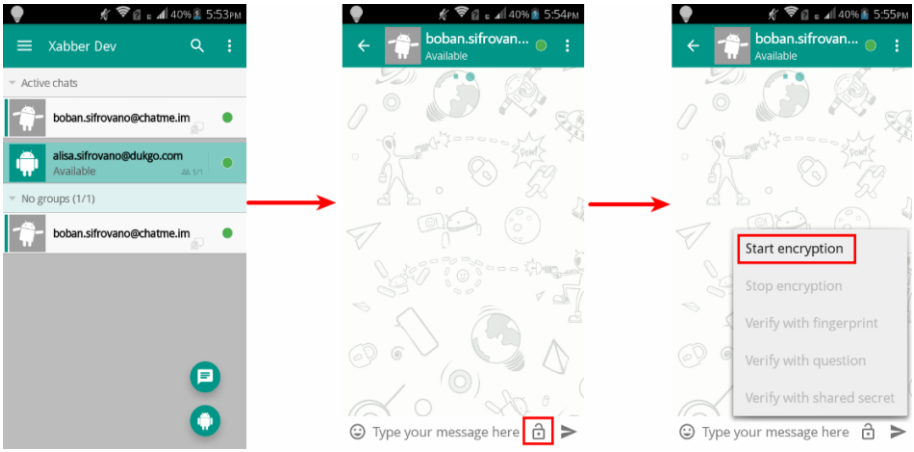
Kada ste dodali vaš nalog, možete dodati nekog vašeg kontakta kako biste sa njim šifrovano ćaskali. Naš kontakt je *boban.sifrovano@chatme.im*. Nakon dodavanja kontakta, vašem kontaktu će stići obaveštenje da ga dodajete i, ako vas odobri, možete ćaskati.



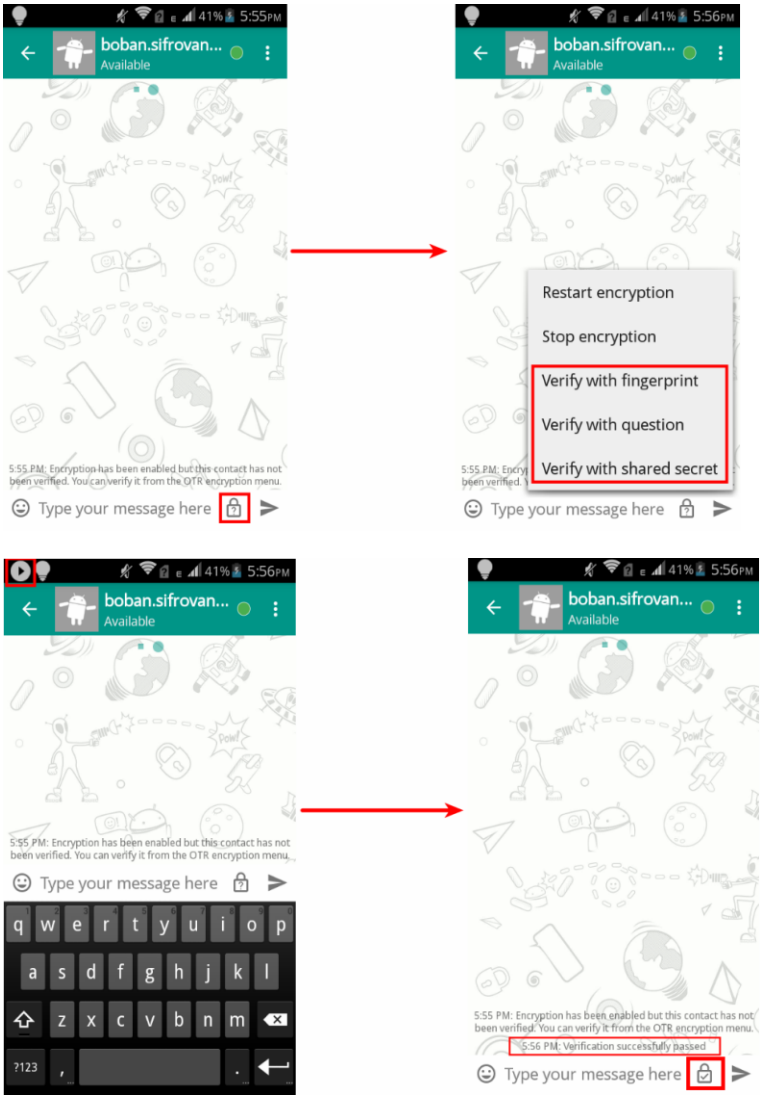
Takođe se može desiti da vas neko doda za svog kontakta i u tom slučaju će vam stići obaveštenje o tome i opcija da to prihvatite ili ne, zavisno od toga da li znate ko je kontakt.

Šifrovano ćaskanje

Nakon što ste dodali kontakta za ćaskanje, mogućnost šifrovanog ćaskanja zavisi od toga da li i vaš kontakt koristi OTR. Naime, Ksaber od korisnika sakriva ove ključeve i svo šifrovanje, pa ako kontakt takođe koristi Ksaber sigurno ćete moći šifrovano komunicirati. Vaš kontakt može koristiti i već pomenute druge XMPP/Jabber klijente ili desktop klijente poput Pidžina, Žicija ili Adiuma, ali unutar tih klijenata mora generisati svoj OTR ključ.



Nakon što ste dodali kontakta možete započeti konverzaciju sa njime. Ali pre nego što išta počnete kuckati, klinite na otključani katanac u donjem desnom uglu, pa „Start encryption”.

Kada ste započeli šifrovanu konverzaciju, potrebno je da verifikujete vašeg sagovornika. To možete učiniti na tri načina, pomoću: poređenja otiska, pitanja i odgovora, ili deljene tajne.

Tek kada i kontakt verifikuje vas (odgovori na pitanje, unese deljenu tajnu ili uporedi otisak), oboje možete biti sigurni da vodite privatnu konverzaciju koju niko ne može presresti i dešifrovati. Videćete u donjem desnom uglu zaključan i štikliran katanac.