Konferencije ETRAN/IcETRAN kroz statistiku

Vladimir A. Katić, *Senior Member, IEEE,* Marko Jarnević, Dragomir Nikolić, Mirjana Jovanić

***Apstrakt*— Konferencija ETRAN je jedan od najstariji naučnih skupova u Srbiji, koji se neprekidno organizuje već 66 godina. Njeno međunarodno izdanje IcETRAN se sada približava prvoj deceniji postojanja. Obe konferencije uspešno organizuje Društvo za ETRAN, čiji kolektivni članovi su sve najznačajnije naučne i obrazovne institucije u Srbiji i Republici Srpskoj (BIH). U radu je prvo predstavljena struktura, kao i ključne teme, koje se na konferencijama razmatraju. Zatim su zajedno posmatrani odgovarajući, karakteristični statistički podaci o broju radova i autora. Oni su obrađivani u tri vremenska intervala, nešto širem (poslednjih 26 god., 1996.-2021. god.), srednjoročnom (poslednjih 9 god., 2014.-2022. god.), i nešto kraćem (poslednjih 4-5 god., 2018. – 2021(2). god.). Brojevi radova su analizirani agregatno, po pojedinačnim konferencijama, ali i detaljnije po tematskim sekcijama, dok su podaci o autorima vezivani za državu i instituciju zaposlenja, kao i za pol istraživača. Zaključeno je da ovi skupovi najčešće predstavljaju mesto prikazivanja naučnih rezultata istraživača iz akademske zajednice (fakulteta i instituta), a da je prisustvo privrede, vojnog i zdravstvenog sektora manje zapaženo. Takođe, najveći broj učesnika je iz Srbije i to sa tri najveća domaća fakulteta. Međutim, značajna je i međunarodna komponenta, kroz autore iz 31 zemlje. Učešće žena je popravljeno, ali još uvek nije adekvatno i iznosi 30%.**

***Ključne reči*— Konferencije ETRAN/IcETRAN, Naučno-stručni radovi, Statistika.**

I. Uvod

Prve inicijative da se organizuje okupljanje inženjera, koji se bave elektronikom pojavile su se još daleke 1953. god. pod rukovodstvom dr Rajka Tomovića. Odbor za elektroniku uradio je sve pripremne aktivnosti i prva konferencija održana je od 7-11. juna 1955. god. [1, 2, 3]. U prvih par godina, konferencija je obuhvatala radove iz elektronike i srodnih disciplina, ali je već 1957. god. osnivanjem Saveznog centra za elektroniku, telekomunikacije i automatiku (ETA) oblast delovanja i zvanično proširena na telekomunikacije i automatiku. Naredne 1958. god. osnovan je Jugoslovenski komitet za ETAN, kada je uz pomenute tri oblasti dodata i nuklearna tehnika (ETAN).

Jugoslovenski komitet, kasnije Jugoslovenski savez za ETAN (od 1976. god.), pa republička Društva za ETAN (od 1980. god.) vodili su konferencije ETAN-a do 1992. god.

Vladimir A. Katić – Predsednik Društva za ETRAN, Univerzitet u Novom Sadu, Fakultet tehničkih nauka, Trg Dositeja Obradovića 6, 21000 Novi Sad, Srbija(e-mail: katav@uns.ac.rs)

Marko Jarnević – Firaso, Kneza Miloša 9/IV sprat, 11000 Beograd, Srbija (e-mail: marko@firaso.rs)

Dragomir Nikolić – Univerzitet u Novom Sadu, Fakultet tehničkih nauka, Trg Dositeja Obradovića 6, 21000 Novi Sad, Srbija (e-mail: nikolicd@uns.ac.rs)

Mirjana Jovanić – Društvo za ETRAN, Kneza Miloša 9/IV sprat, 11000 Beograd, Srbija (e-mail: etran.konferencija@gmail.com)

Društvo za elektroniku, telekomunikacije, računarstvo, automatiku i nuklearnu tehniku (Društvo za ETRAN), osnovano je 1993. god., kada je u tematske sadržaje konferencije uključeno i računarstvo, kao posebna sekcija [1]. To je rezultiralo promenom naziva iz ETAN u ETRAN. Društvo je nastavilo sa širenjem delatnosti i već 1994. god. je osnovana sekcija za elektroenergetiku (EE), a 1997 sekcija za metrologiju (ML).

Potreba za širom vidljivošću rezultata istraživanja u Srbiji dovela je do pokretanja međunarodne konferencije, IcETRAN (*International Conference on Electrical, Electronic and Computing Engineering*) 2014. god. sa istim tematskim okvirom, a na kojoj se predstavljaju radovi na engleskom jeziku u IEEE formatu. Od tada se konferencije ETRAN i IcETRAN odvijaju zajedno (paralelno) u istim terminima i na istoj lokaciji.

Trenutno ETRAN ima šesnaest sekcija, kojima su pokrivene sve moderne oblasti elektrotehnike. Karakteriše ga veliki broj individualnih, ali i kolektivnih članova (28 institucija i organizacija iz Srbije, Bosne i Hercegovine i Crne Gore). Ove 2022. god. održava se LXVI konferencija ETRAN-a, odnosno IX konferencija IcETRAN-a.

U literaturi se retko pojavljuju radovi u kojima je urađena statistička obrada naučne produkcije konferencija ETA, ETAN, ETRAN i IcETRAN. U publikaciji [1], dat je prikaz perioda od 2006. – 2015. god., s tim da su pojedine sekcije dale i celokupan pregled kretanja broja radova od svog osnivanja do 2015. god., dok je u [2] dat prikaz ranijih konferencija (1955-2014. god.), s tim da statistika nedostaje za period 1985-1996. god. U [4] je dat reprint svih radova u periodu 1955. – 2006. god., ali nema agregiranog statističkog prikaza, kao ni dela radova iz zbornika 1992. i 1993. god. i nekih ostalih godišta.

Neki podaci mogu se dobiti i iz uvodnih referata podnesenih na konferencijama ETRAN-a. Tako je na XL konferenciji u Budvi u junu 1996. god. prof. Milić Stojić, tadašnji predsednik ETRAN-a napisao: „Na proteklih 39 konferencija bilo je podneto ukupno 10.021 rad, koji su zatim publikovani na oko 70.000 stranica u 187 tomova zbornika radova. Na jubilarnoj XL Konferenciji očekuje se da će biti podneto 624 rada 1156 autora u okviru 15 stručnih komisija i 67 radnih sednica.“ [5].

Na sajtu ETRAN/IcETRAN konferencija dati su reprinti kompletnih zbornika sa poslednjih pet konferencija (2017.-2021. god.), a unos ranijih godišta je u toku [6]. Zbog kratkoće vremena i teškoće nabavke, autori ovog rada nisu bili u prilici da konsultuju ostalu literaturu vezanu za aktivnosti društva za ETAN/ETRAN, što ostaje kao zadatak za naredni period.

Cilj ovog rada je da predstavi sumirani pregled tematike i

 ISBN 978-86-7466-930-3

aktivnosti na obe konferencije (nacionalne i međunarodne) u tri vremenska intervala, nešto širem (poslednjih 26 god., 1996.-2021. god.), srednjoročnom (poslednjih 9 god., 2014.-2022. god.), i nešto kraćem (poslednjih 4-5 god., 2018. – 2021(2). god.), koristeći različite statističke prikaze broja radova i autora. Na taj način želi se ukazati na opšte pravce razvoja, na njihove trendove, ali i na segmente kojima je potrebno posvetiti veću pažnju i trud u budućnosti.

II. KONFERENCIJE ETRAN/ICETRAN

Konferencije ETA/ETAN/ETRAN organizuju se godišnje, počevši od 1955. god. Prva i druga konferencija održane su u Beogradu 1955. i 1956. god., a nadalje su menjale lokacije, širom bivše Jugoslavije. Od 1992. god. održavaju se u Srbiji i Crnoj Gori, a od 2008. god. isključivo u Srbiji [2]. Od 2014. god. društvo za ETRAN svake godine organizuje dve paralelne konferencije, nacionalnu ETRAN i međunarodnu IcETRAN u nekom mestu u Srbiji. Najviše domaćinstava do sada ima Beograd 7 puta, Zlatibor 6 puta, pa Novi Sad i Niš po 4 puta. Ove 2022. god. ETRAN je prvi put u Novom Pazaru, kao mladom univerzitetskom centru.

Rad ETRAN/IcETRAN konferencija odvija se preko 16 sekcija, koje pokrivaju kompletnu oblast elektrotehnike i predstavljaju ključne teme svih konferencija. To su (po abecednom redu): Akustika (AK), Antene i prostiranja (AP), Automatika (AU), Biomedicinska tehnika (BT), Električna kola, električni sistemi i obrada signala (EK), Elektroenergetika (EE), Elektronika (EL), Metrologija (ML), Mikroelektronika i optoelektronika (MO), Mikrotalasna tehnika, tehnologije i sistemi (MT), Novi materijali (NM), Nuklearna tehnika (NT), Računarstvo (RA), Robotika i fleksibilna automatizacija (RO), Telekomunikacije (TE) i Veštačka inteligencija (VI).

Pored njih, na konferencijama se pojavljuju i radovi iz oblasti nastavne problematike, odnosno edukacije (EDU), kao i oni koji tematski obuhvataju interdisciplinarne oblasti,ili tematiku više struka, a koji su svrstani u specijalne sesije.

Na svakoj konferenciji, nekoliko radova ili predavanja se predstavljaju na plenarnim sednicama, kao uvodni ili *Key Note Lectures*. Takođe, u sklopu svake sekcije, prezentuju se pozvani radovi (*Invited Papers*) ili pozvana predavanja (*Invited Lectures*).

Uz prezentaciju radova, na konferencijama organizuju se i okrugli stolovi (*Pannel Sessions*) vezani za aktuelnu problematiku razvoja nauke u svetu ili Srbiji, kao i sednice posvećene pojedinim autorima.

III. STATISTIKA BROJA RADOVA

Pregled broja radova po svim dosadašnjim konferencijama, kako ukupan, tako i po pojedinim sekcijama, prevazilazi okvire ovog rada, jer zahteva znatno više prostora za dijagrame, objašnjenja i komentare. Da bi se ipak dobio odgovarajući prikaz, autori su odlučili da se ograniče na statistiku broja radova u tri pomenuta vremenska intervala. Treba napomenuti da su za konferencije od 1996. god. do 2021. god. u statistiku uvršteni radovi, koji su ušli u Zbornik/Proceedings (prezentovani radovi), dok su za 2022. god. obuhvaćeni prihvaćeni radovi, koji su prošli duplu recenziju.

Na Sl. 1 prikazan je pregled kretanja broja radova na konferencijama ETRAN i IcETRAN u ovom periodu. Može se videti da je kvantitativno konferencija ETRAN krajem devedesetih godina prošlog veka predstavljala jedan od najvećih, ako ne i najveći naučni skup u Srbiji. Prema podacima za 1996. i 1997. god., XL i XLI konferencija održane u junu u Budvi i na Zlatiboru, respektivno, predstavile su čak 624, odnosno 622 naučna i stručna rada [2, 4].

U narednom periodu, broj radova polako opada, tako da se može zapaziti nekoliko kvantitativnih blokova vezanih za kretanje broja radova:

1996. – 1998. god.: Između 500 – 600+ radova, prosek 588,6
1999. – 2010. god.: Između 300 – 500 radova, prosek 352,5
2011. – 2019.god.: Između 200 – 300, prosek 269,3
2020. – 2022.god.: Između 100 – 200, prosek 158,3

Poslednji blok ukazuje da je sada broj radova dosta nizak, odnosno da odgovara nivou iz početnog perioda razvoja konferencije. Tek 1975. god. na XIX konferenciji u Ohridu broj radova je prešao obim od 200 radova i tu se zadržao sve do 2019. god.Ipak, stanje za konferenciju 2022. god. ohrabruje, jer je broj radova značajno povećan (+36%).

Međutim, sadašnje stanje nije na nivou reputacije konferencije(a) i upućuje da je potrebno uložiti dodatne napore na povećanju atraktivnosti konferencije, kvaliteta radova, širem povezivanju sa privredom i snaženju uticaja u naučnoj i stručnoj javnosti Srbije i sveta.

Sl. 1. Sveobuhvatni pregled broja radova na ETRAN/IcETRAN konferencijama 1996.-2022. god.

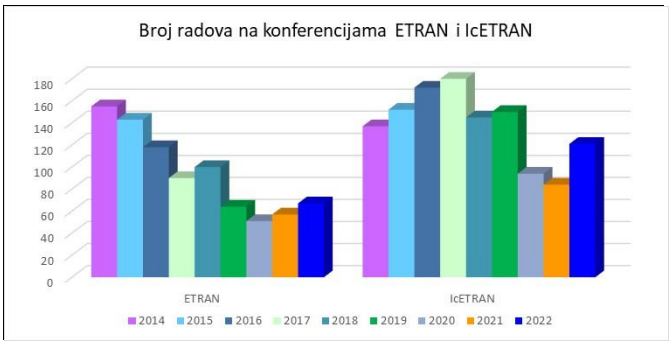
 ISBN 978-86-7466-930-3

A. Statistika radova 2014. – 2022. god.

Iz razloga aktuelnosti i ovde će se posebno analizirati period 2014. – 2022. god. Karakteristika tog perioda je da se u njemu pojavljuju dve paralelne konferencije ETRAN, kao nacionalna i IcETRAN, kao međunarodna. To daje mogućnost autorima da svoje radove predstave na srpskom ili engleskom jeziku. Pored već pomenutih razloga internacionalizacije i bolje vidljivosti u inostranstvu, razlog za ovakvu odluku predsedništva ETRAN-a može se tražiti i u sve većem učešću naučnika iz Srbije na međunarodnim projektima, interesovanju svetskih naučnika za rezultate u Srbiji, intenziviranju značaja citiranosti za rangiranje naučnih radnika, ali i kao način snaženja uticaja i značaja konferencije u Srbiji i svetru.

Na Sl. 2, prikazan je „zumiran“ pregled broja radova sa slike 1, na obe konferencije u analiziranom periodu, ali prikazan odvojeno, odnosno uporedno po konferencijama. Ukupno je na konferenciji ETRAN predstavljeno 847 radova, dok na konferenciji IcETRAN 1239 radova, odnosno sve zajedno na obe konferencije 2086 radova. I ovde se može uočiti napredak u 2022. god.

Poređenjem dve konferencije, vidi se da se većina autora opredeljuje za konferenciju IcETRAN (59,4%), što ukazuje na potrebu dalje internacionalizacije konferencije. Ipak, značajan broj autora prikazuje svoje rezultate i na ETRAN-u, kao nacionalnom skupu, što je dobro jer podstiče razvoj domaće misli, usvajanje adekvatne terminologije na srpskom jeziku, odnosno umanjuje efekte anglikanizacije.



Sl. 2. Pregled broja radova na konferencijama ETRAN/IcETRAN u periodu 2014. – 2022. god.

B. Statistika radova po sekcijama (2018. – 2022. god.)

Nešto bolji uvid u rezultate istraživanja može se dobiti ako se posmatra broj radova po pojedinim sekcijama. Međutim, s obzirom na broj sekcija, predstavljanje pomoću jedinstvenih dijagrama može biti nepregledno. Iz tog razloga, autori ovog rada odlučili su se da dodatno suze opseg posmatranja na poslednjih pet godina, odnosno na period od 2018. do 2022. god.

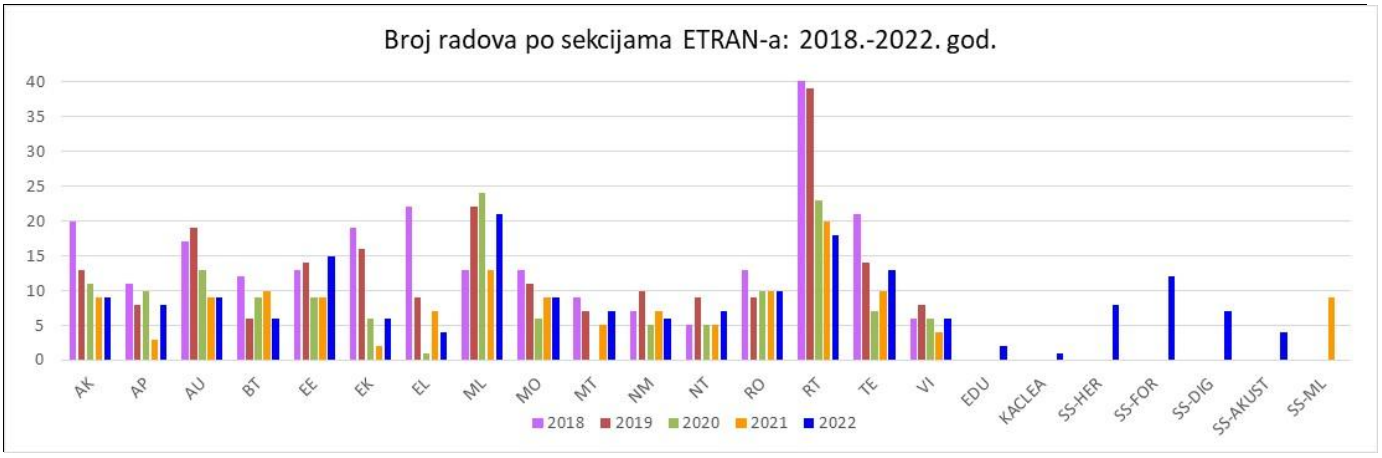
Na Sl. 3 predstavljen je broj radova po sekcijama u tom periodu. Vidi se da su najistaknutije bile sekcije za računarstvo (RT) i metrologiju (ML), zatim automatiku i obradu signala (AU), telekomunikacije (TE), akustiku (AK) i elektroenergetiku (EE). Kod nekih sekcija može se videti značajna redukcija aktivnosti (na primer, sekcije AK, AU, EK, EL, RT), kod nekih da su konferencije u 2020. ili 2021. god. bile kritične (na primer AP, EL, EK, MT), ali se kod većine zapaža oporavak u 2022. god. Ukupnom pozitivnom rezultatu za 2022. god. dodatno doprinose radovi u EDU sesiji i specijalnim sesijama, kao i predstavljanje međunarodnog projekta KALCEA.

C. Statistika radova po institucijama

Na skupovima ETRAN/IcETRAN učestvuje veliki broj autora, najčešće sa univerziteta (fakulteta), naučnih instituta, privrede, ali i iz vojno-tehničkih i ustanova zdravstva. Naravno, mogući su i istraživački timovi kombinovani sa članovima iz različitih institucija. Kompletno sagledavanje prevazilazi obim ovog rada, pa je fokus stavljen na period 2018. – 2021. god.

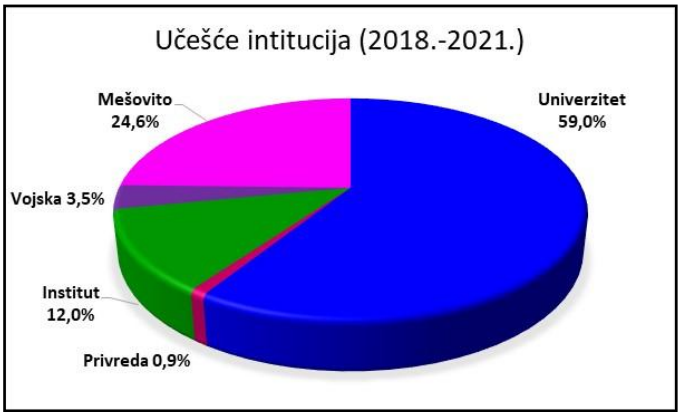
Na Sl. 4 predstavljeno je učešće broja radova po grupisanim institucijama. Vidi se da su radovi autora sa univerziteta (fakulteta) bili najviše zastupljeni (59,0%), pa zatim radovi mešovitih timova (24,6%), istraživačkih grupa sa naučnih instituta (12,0%), te vojno-tehničkih ustanova (3,5%) i privrede (0,9%). Interesantno je da nije bilo većeg učešća radova autora isključivo iz zdravstvenih ustanova, odnosno oni su se pojavljivali u sklopu mešovitih timova.

Ovaj pregled pokazuje da su skupovi ETRAN/IcETRAN prvenstveno orijentisani na učesnike iz akademskog okruženja (fakulteta ili instituta), a da je prisustvo autora iz privrede i drugih institucija nedovoljno.



Sl. 3. Pregled broja radova po sekcijama u periodu 2018.-2022. god.

 ISBN 978-86-7466-930-3



Sl. 4. Pregled učešća radova po institucijama za period 2018. – 2021. god.

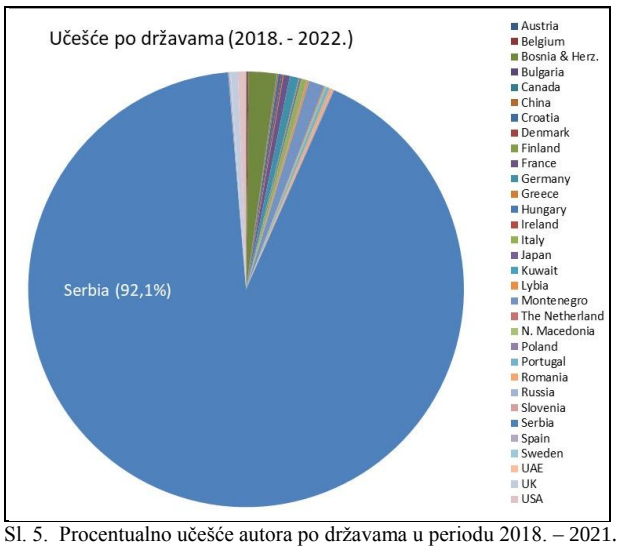
IV. STATISTIKA PO AUTORIMA

Zbog znatno složenije obrade, za statistiku po autorima koristiće se ograničeni vremenski interval od 4 odnosno 5 godina (2018. – 2021(2). god.). Na bazi iskustva i prethodnih podataka, može se odmah pretpostaviti da autori konferencijskih radova, odnosno učesnici ETRAN/IcETRAN konferencija uglavnom dolaze iz Srbije i iz akademskih institucija. Međutim, statistika ukazuje da su ove konferencije interesantne i za mnoge učesnike iz inostranstva i iz drugih institucija. Iz tih razloga interesantno je posmatrati i uporediti konferencije po nekim parametrima vezanim za autore. Ovde će se predstaviti statistika po državama iz kojih dolaze autori, odnosno u kojima se nalaze institucije i firme gde su zaposleni, zatim po pojedinačnim institucijama (mestu zaposlenja) autora, te po polu da bi se videla kakva je zastupljenost žena.

A. Autori iz inostranstva

Da bi se potvrdila pomenuta pretpostavka, na Sl. 5 prikazano je procentualno učešće autora iz Srbije i iz drugih država u razmatranom periodu. Može se uočiti da velika većina, čak 92,1% dolazi iz Srbije. To je i očekivano, zbog nacionalnog karaktera konferencije ETRAN, ali i zbog toga što značajan broj autora je iz Srbije publikuju svoje rezultate na engleskom jeziku u okviru konferencije IcETRAN.

Ipak, detaljniji pregled autora pokazuje da preostalih 7,9% predstavljaju institucije iz čak 31 zemlje sveta. Da bi se jasnije videla njihova zastupljenost, na Sl. 6 predstavljena je statistika autora iz inostranih institucija. Može se uočiti da ih najviše dolazi iz Bosne i Hercegovine (uglavnom Republike

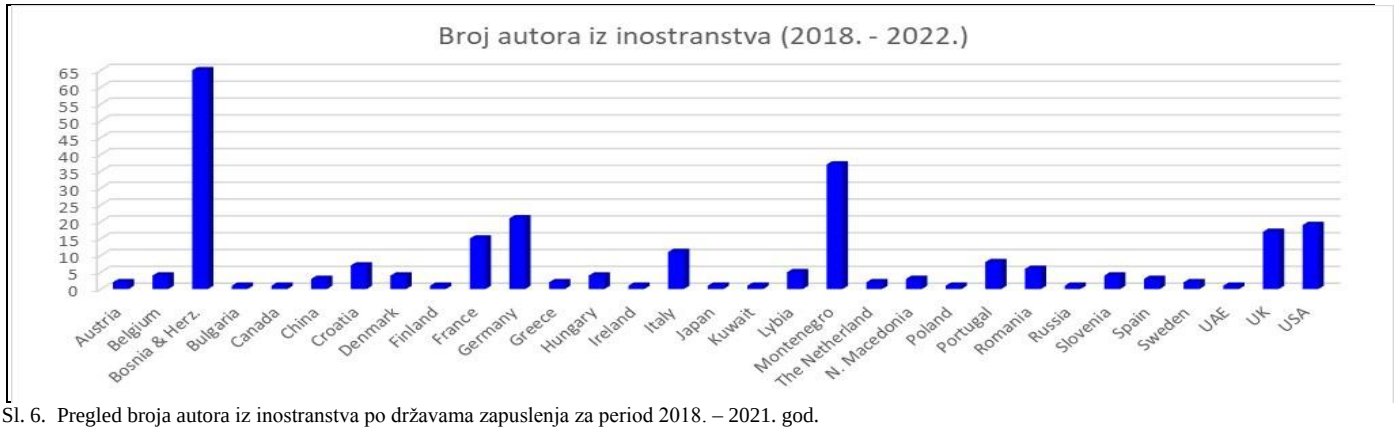


Sl. 5. Procentualno učešće autora po državama u periodu 2018. – 2021. god.

Srpske) i Crne Gore, pa zatim iz Nemačke, Velike Britanije (UK) i Sjedinjenih Američkih Država (USA). To ukazuje na dobru saradnju u regionu, jer su i autori iz Hrvatske, Slovenije i Severne Makedonije prisutni. Treba napomenuti da razlog za manje učešće tri poslednje pomenute države ex-Jugoslavije se može potražiti i u činjenici da one imaju svoje nacionalne konferencije, kojhe su potekle iz ETAN-a: KoREMA (bivša JUREMA), ELMAR (bivši ETAN u pomorstvu), ETAI i dr. Istaknuto prisustvo autora iz razvijenih, zapadnih zemalja je najvećim delom posledica aktivnosti naše naučne dijaspore, bilo u sklopu zajedničkih istraživanja sa autorima iz Srbije, bilo u sklopu timova sa svojih institucija.

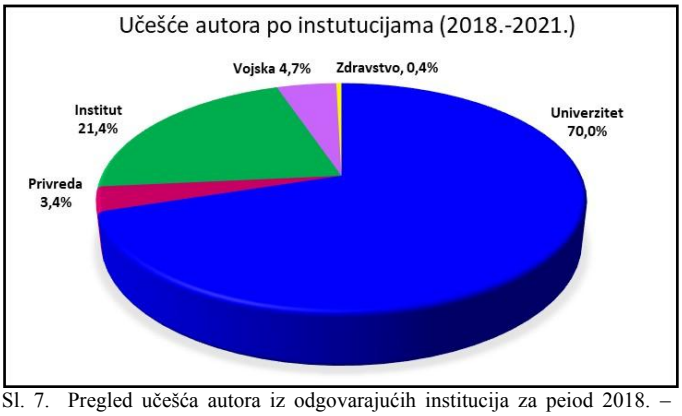
B. Pregled po institucijama autora

Već je pokazano da najviše radova dolazi od autorskih timova sa univerziteta ili instituta, ali da je značajno i učešće kombinovanih timova (Sl. 4). Iz tog pregleda se ne vidi jasno pripadnost institucijama pojedinačnih autora, pa je na Sl. 7 dat pregled učešća autora po institucijama grupisanim kao na Sl. 4 za period 2018. – 2021. god. Naravno, opet je najznačajnije prisustvo autora sa univerziteta (fakulteta) 70,0%, instituta 21,4%, ali se vidi i primetno učešće autora iz vojske 4,7%, privrede 3,4% i zdravstva 0,4%. Poređenjem sa Sl. 4 može se zaključiti da su radovi mešovitih timova najčešće rezultat saradnje istraživača sa univerziteta, odnosno instituta i privrede ili vojske, i nešto manje zdravstva.



Sl. 6. Pregled broja autora iz inostranstva po državama zapuslenja za period 2018. – 2021. god.

 ISBN 978-86-7466-930-3



Sl. 7. Pregled učešća autora iz odgovarajućih institucija za peiod 2018. – 2021. god.

Pored pregleda po grupisanim institucijama, interesatno je analizirati mesta zaposlenja autora, tj. napraviti pregled učešća pojedinačnih institucija. Na Sl. 8 je dat ovakav pregled za period 2018.-2021. god., uz napomenu da su u obzir uzete samo one institucije čiji autori se pojavljuju bar 3 puta. Vidi se da je najveće učešće na konferencijama ETRAN/IcETRAN od strane autora sa Fakulteta tehničkih nauka iz Novog Sada (FTNNS), zatim sa Elektrotehničkog fakulteta u Beogradu (ETFBG), te Elektronskog fakulteta u Nišu (EFNI). Zatim ide Institut RT-RK iz Novog Sada (RTRK), Visoka škola elektrotehnike i računarstva strukovnih studije Beograd (VSEL), Fakultet tehničkih nauka iz Čačka (FTNCA), te grupa instituta: Institut za hemiju, tehnologije i metalurgiju iz Beograda (IHTM), Institut za nuklearne nauke Vinča (INNVIN), Institut Vlatacom Beograd (VLATA) i Institut Mihajlo Pupin iz Beograda (IMPBG).

Ovaj pregled potvrđuje da su skupovi ETRAN/IcETRAN prvenstveno orijentisani na učesnike iz akademskog okruženja, odnosno na autore sa fakulteta i instituta u Beogradu, Novom Sadu, Nišu i Čačku, ali je značajna i prisutnost autora iz specijalizovanih firmi (industrije), kao i iz inostranstva. Međutim, prisustvo autora iz privrede, te saradnja sa zdravstvom još uvek je nedovoljna.

C. Pregled broja autora prema polu

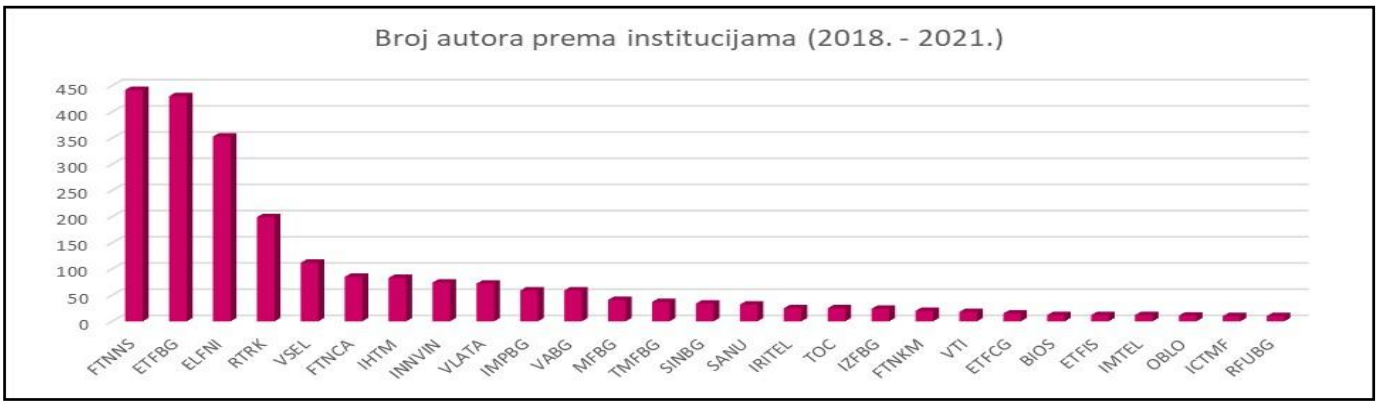
U poslednje vreme sve više se govori i ulažu se dodatni napori da se poveća broj žena u inženjerskim strukama. Njihovo učešće postaje sve značajnije, pa je interesantna i statistika o njihovoj participaciji u naučnim radovima, koji se izlažu na ETRAN/IcETRAN konferencijama. Na Sl. 9 prikazan je pregled procentualnog učešća muškaraca i žena na ovim konferencijama u periodu 2018.-2021. god. Može se uočiti da je došlo do blagog povećanja učešća žena, odnosno da je ono sa 23,7% na konferencijama 2018. god. dostiglo 30% na konferencijama 2021. god. To je ohrabrujući trend, ali treba ulagati dodatne napore da se on održi.

V. ZAKLJUČAK

Konferencija ETRAN, a sada i IcETRAN, u svom dugom postojanju prošle su kroz razne faze. Poslednjih godina uočljiv je rapidan pad broja radova i smanjeno interesovanje naučnika i istraživača. Šta su razlozi i uzroci ove negativne pojave svakako zahteva dublje analize, ali se oni mogu tražiti u sveukupnoj digitalizaciji naučne produkcije, pa i svih ostalih sfera rada i života, u specifičnim zahtevima vrednovanja naučnog rada gde je fokus stavljen na naučne časopise, u potrebi za većom dinamikom naučnog rada i produkcije, pa do uticaja ograničenja u finansiranju nauke, negativnim efektima pandemije virusa Korona-19 i dr.

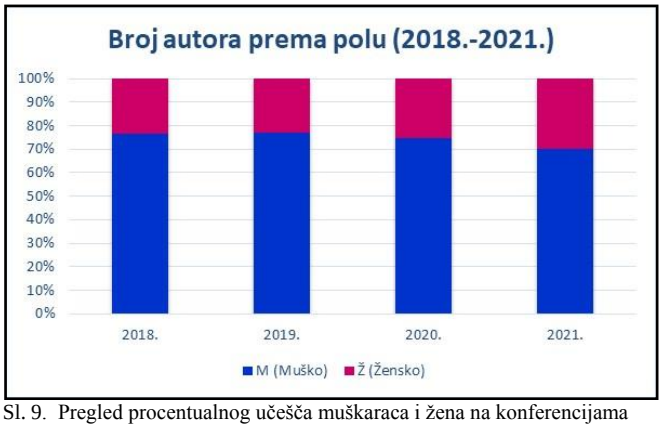
Ovi skupovi najčešće predstavljaju mesto prikazivanja naučnih rezultata istraživača iz akademske zajednice (fakulteta i instituta), a prisustvo privrede, vojnog i zdravstvenog sektora treba intenzivirati. Takođe, najveći broj učesnika je iz Srbije i to sa tri najveća domaća fakulteta. Međutim, značajna je i međunarodna komponenta, kroz autore iz čak 31 zemlje. Učešće žena se popravlja, ali još uvek nije adekvatno i sada iznosi 30%.

Ovogodišnje konferencije, LXVI ETRAN i IX IcETRAN nagoveštavaju preokret i bolju budućnost ovih okupljanja. Broj radova značajno je porastao (+33%), kao i očekivano učešće istraživača. Konferencija je ponovo organizovana „u živo“, a postavljeni su osnovi za intezivnije pojavljivanje *hi-tech* privrede i veze sa naučno-tehnološkim parkovima. To ukazuje da osnovni razlozi okupljanja naučnika i istraživača, a to je neposredna razmena rezultata i iskustava, dolaženje do novih ideja i zajedničko druženje i dalje predstavljaju značajan motiv za učestvovanje. Ovo i ohrabruje na nove napore za poboljšanje značaja i organizacije konferencija, povećanje atraktivnosti, uvećanje učešća naučnika, istraživača i stručnjaka iz privrede, kao i mladih doktoranata, internacionalizaciju, povezivanje sa svetskim asocijacijama (IEEE, IFAC i dr.), kao i bolju vidljivost ovih konferencija u svetskoj i domaćoj naučnoj zajednici.



Sl. 8. Pregled broja autora prema institucijama za period 2018. – 2021. god.

 ISBN 978-86-7466-930-3



Sl. 9. Pregled procentualnog učešća muškaraca i žena na konferencijama ETRAN/IcETRAN u periodu 2018.-2021. god.

LITERATURA

[1] ***, *„ET(R)AN Prvih šezdeset konferencija – Doprinos ravoju elektrotehničke struke“*, Uredili: B. Milovanović i Z. Jakšić, pp. 90-98, Beograd, Društvo za ETRAN i Akademska misao, 2016.

[2] M. R. Stojić, *„Društvo za ETAN/ETRAN 1953-2016“*, Akademska misao, Beograd, 2016.

[3] S. Ristić, „Dvadeset godina ETAN-a“, XVII konferencija ETAN, Novi Sad, jun 1973.

[4] B. Milovanović, B. Kovačević, Z. Jakšić, M. Jovanić i dr., *„e-TRAN Elektronski zbornik svih radova prvih pedeset konferencija ETAN/ETRAN 1955-2006“*, Beta verzija, DVD , Društvo za ETRAN, Beograd, 2006.

[5] M.R. Stojić, „Uticaj četrdeset konferencija ETRAN-a na razvoj naučne i stručne misli“, Plenarni rad po pozivu, Zbornik sa XL Konferencije za ETRAN, Budva, 1996., pp.3-6.

[6] https://www.etran.rs

ABSTRACT

The ETRAN conference is one of the oldest scientific gatherings in Serbia, which has been organized continuously for 66 years. Its international edition IcETRAN is now approaching its first decade of existence. Both conferences are successfully organized by the Society for ETRAN, whose collective members are all the most important scientific and educational institutions in Serbia and the Republika Srpska (BIH). The paper first presents the structure, as well as key topics, which are discussed at conferences. Then, the corresponding, characteristic statistical data on the number of papers and authors were observed together. They were analyzed for three periods, a broader one (last 26 years, 1996-2021), a medium-term (last 9 years, 2014-2022), and a shorter one (last 4-5 years, 2018 – 2021(2)). The numbers of papers were analyzed in aggregate, but also by individual conferences and in more detail by thematic sections, while the data on the authors were related to the country and the institution of employment, as well as to the gender of researchers. It was concluded that these gatherings usually represent the place of presenting the scientific results of researchers coming from the academia (faculties and institutes) and that the presence of the ones from industry, military, or healthcare is less noticeable. Also, the largest number of participants is from Serbia, coming from the three largest national universities. However, the international component is also significant, through authors from 31 countries. The participation of women has been improved, but it is still not adequate and amounts to 30%. These data were the fundamentals to discuss the future development of both conferences.

The ETRAN/IcETRAN ConferencesThrough Statistics

Vladimir A. Katić, Marko Jarnević, Dragomir Nikolić, Mirjana Jovanić

 ISBN 978-86-7466-930-3

A Comparison of Selected Systems For Learning About SQLi Vulnerability Suitable for Academic Uses

Djordje Madic, Danko Miladinovic and Zarko Stanisavljevic

***Abstract*— In this paper five popular platforms for secure software development training are analyzed from the perspective of their suitability for academic uses. In order to compare these platforms a novel taxonomy of interactive cyber training and education systems (Cyber Taxi) is used. Only parts of the taxonomy that are relevant are included in the analysis. The analyzed platforms are also compared to SQLiTrainer system, which was developed at the University of Belgrade, School of Electrical Engineering specifically to be used at the courses dealing with the SQL injection (SQLi) vulnerability. Based on the conducted analysis a suggestion is made regarding the requirements that a training platform should fulfill in order to be suitable for academic uses.**

***Index Terms*— SQLi; secure software development; hands-on training.**

I. Introduction

Software engineering never had more learning resources than today, with the internet resources available at all times. It comes in many forms, as video courses, articles, e-books, and most of them are free. Most of the ones covering SQLi are structured to build theoretical knowledge while applying the knowledge and hands-on experience are missing.

On the other hand, courses covering topics like secure software development and network and system security are emerging across universities all around the world. Respectable authorities in the field of software engineering, i.e. ACM and IEEE, are issuing their recommendations regarding curriculum [1] and in these recommendations, as a rule, they state that courses covering topics from software engineering should be supported by hands-on experience for students.

There is a constant dilemma when introducing hands-on exercises at the university course, should this be done using existing technology or is there a need for a new dedicated tool to be created. In order to help teachers to resolve this dilemma regarding SQLi vulnerability as a topic to be covered at some course, in this paper analysis of five popular platforms for secure software development training is presented. These platforms are also compared to one dedicated tool for teaching SQLi vulnerability as a course topic (SQLiTrainer [2]). Based on qualitative analysis, a suggestion of requirements that a tool suitable for academic uses should fulfill is made. Finally, an example of quantitative analysis is presented. As far as the authors are aware, there are no similar studies published in open literature.

The remainder of the paper is organized as follows. The second section presents the three selected tools with hands-on SQLi exercises, one online tool with hands-on SQLi exercises, one framework for organizing secure software training, and one dedicated teaching tool. The third section compares the six selected tools using the taxonomy of interactive cyber training and education systems (Cyber Taxi) [3]. The section four describes the requirements that a tool should fulfill in order to be used in academia. The fifth section gives an example of quantitative comparison of the selected tools. The last section gives a conclusion and suggests future work.

II. Description Of The Selected Tools

Five popular platforms are selected for this research based on their availability to the authors, their popularity in the secure software development community and their coverage of the selected topic (e.g., Juice Shop [4] is an official tool of the OWASP [5] community, Avatao [9] is used for the Serbian Cyber Security Challenge, etc.).

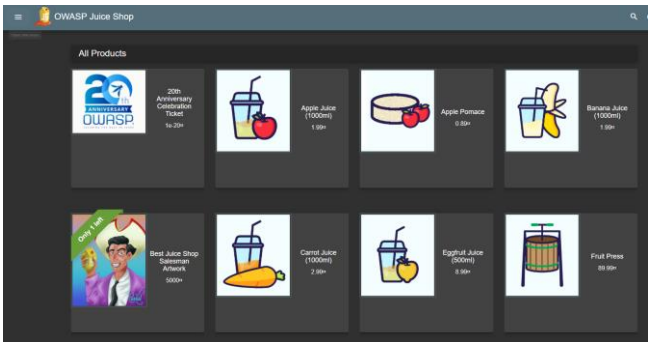


Fig. 1. OWASP Juice Shop

Juice Shop [4] (Fig. 1) is an open-source web application representing an insecure online shop. It is created by OWASP [5] organization and contains 100 challenges of varying difficulty where the user is supposed to exploit the underlying

Djordje Madic is with Zuehlke Engineering, Bul. Milutina Milankovića 1i, 11070 Novi Beograd, Serbia (e-mail: djordje.madic@zuehlke.com).

Danko Miladinovic is with University of Belgrade, School of Electrical Engineering, Bul. kralja Aleksandra 73, 11120 Belgrade, Serbia (phone: +381-63-3439-97; e-mail: danko@etf.bg.ac.rs).

Zarko Stanisavljevic is with University of Belgrade, School of Electrical Engineering, Bul. kralja Aleksandra 73, 11120 Belgrade, Serbia (phone: +381-11-3218-484; e-mail: zarko.stanisavljevic@etf.bg.ac.rs).

 ISBN 978-86-7466-930-3

vulnerabilities. SQL injection is covered by 7 of them. The application automatically detects when a challenge is solved, and progress of the user is tracked on a score board. All components of the user interface can be customized including color theme, logos, banners, links, products and other items in the database. Interactive help, hints and „challenge solved“ notifications can be turned off, while the initial set of challenges and their solutions cannot be changed. Domain of the application is well chosen, having in mind the number of online shops today. According to a study from 2017 [6] there are 800.000 registered online shops only in Europe. Juice Shop application code [7] has more than 50 contributors. Instance of the application can be started on a personal computer or one of the cloud services.

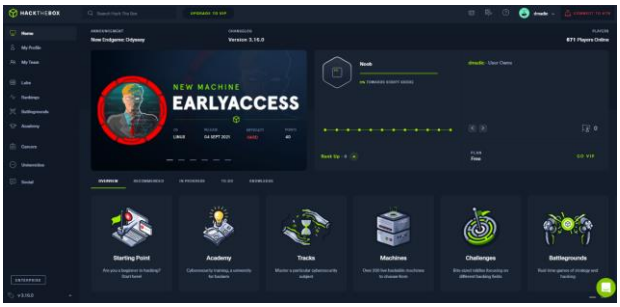


Fig. 2. Hack The Box

Hack The Box [8] (Fig. 2) is online training platform, helping individuals, companies and universities improve penetration testing skills. The platform is free for students and university professors. It includes a set of Capture The Flag (CTF) exercises grouped in 10 categories, like Web, Hardware and Reverse Engineering. Exercises are updated on weekly basis. Progress of the user is tracked and awarded with points, ranks and badges. Exercises are performed in a realistic environment, for example, against a dedicated web application instance or an executable that should be reverse engineered. Part of the exercises is followed by detailed documentation of the solutions. At the moment of writing, Web category contains 34 challenges designed as complex attacks where the user has to exploit multiple vulnerabilities to solve them. Some of them require SQLi to be solved.

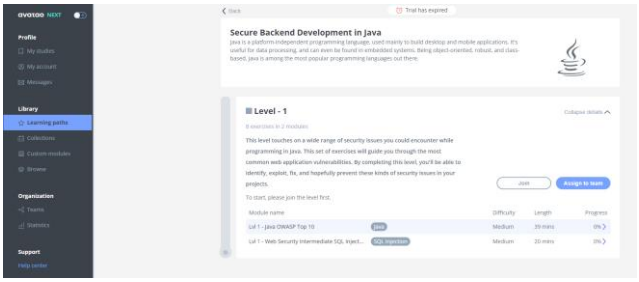


Fig. 3. Avatao

Avatao [9] (Fig. 3) is online training platform for companies created to improve security awareness of the employees and provide them with best practices for secure software development. It includes exercises in 8 programming languages, grouped in modules and learning paths. Exercises are performed in a realistic environment, for example, against a dedicated web application instance. There are 2 types of exercises, challenges and tutorials. Tutorials are interactive exercises guided by a chat bot acting as a teacher, while the challenges are designed as CTF. Most of the exercises are focused on single vulnerability, like SQLi. There are both attack and prevention oriented exercises. At the moment of writing, trial access to the platform provides 18 SQLi exercises in 5 programming languages. Many of them are different variants of SQLi Login Bypass.

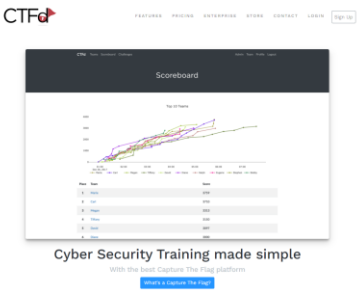


Fig. 4. CTFd Admin panel

CTFd [10] (Fig. 4) is online training framework for hosting CTF competitions and is used for helping individuals and companies improve cyber security skills using CTF challenges. Beside CTF challenges the platform supports a variety of other types of challenges such as multiple-choice and manual verification exercises. Some of these types of challenges are not available in the free version of the framework. Having said that, the framework is free, and it can be expanded by buying plugins and themes from the CTFd website. Unlike other tools, challenges do not come with the framework, but they can be added from the admin page. The admin can also add more pages, view the progress of individual users and teams on the platform and view success rate of each challenge. The framework supports individual and team competitions.

Fig. 5. PicoCTF

PicoCTF [11] (Fig. 5) is online training tool for helping users increase their hacking skills using CTF challenges. Users can register either as learners or as teachers. Learners can solve CTF challenges and join a classroom. Upon joining classrooms learners can get custom event scoreboards and track classroom member’s progress. Teachers can create

 ISBN 978-86-7466-930-3

classrooms and monitor student progress. Exercises are CTF based and are split into categories like web exploitation, cryptography, forensics, etc. There is no SQLi category, but there are around 10 SQLi challenges across all categories.

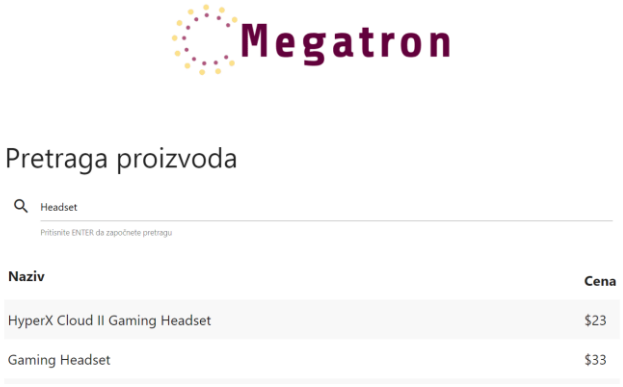


Fig. 6. SQLiTrainer

SQLiTrainer [2] (Fig. 6) represents a set of 4 vulnerable applications that can be used to demonstrate different types of SQL injection vulnerabilities. Implementation and the examples how to use the system are given in [2]. It is used for laboratory exercises at the Advanced Network and System Security course at the University of Belgrade, School of Electrical Engineering. Components of the user interface and initial state of the database can be customized. The applications can be run against different SQL databases, like MySQL and PostgreSQL, changing the solution of the exercises.

III. Comparison Of The Selected Tools

In order to compare different tools a common ground and a set of measurable parameters are needed. For the purpose of suggesting requirements that a tool should fulfill to be used in academia a novel and promising taxonomy Cyber Taxi [3] was analyzed. Most relevant parts of the taxonomy are orchestration, proficiency level, and customization level. In [3] orchestration defines the automation level, although this term can have other meanings in a different context. Having a group of students, it is a requirement to be able to set up the training environment automatically, no matter the number of participants. In case of problems, it should be simple to restart the environment and debug it. Proficiency level defines required level of knowledge. In case it is above beginner level, the tool is not relevant for academic use. In order to provide enough learning material and assess different groups of students, training should be customizable, and should be able to produce similar exercises.

Classification of the selected tools based on the relevant parts from the Cyber Taxi is presented as a table in Fig. 7. Columns of the table represent the selected tools, while rows of the table represent components of the taxonomy, grouped by more general concepts. The table is filled based on the experience of the authors with the selected tools using their publicly available, free or trial versions. All of the tools except CTFd have trainings of similar purpose and include exercises for beginners. SQLiTrainer is the only tool requiring manual effort for scoring. Hack The Box, Avatao and picoCTF are ready-to-use products, while Juice Shop, SQLiTrainer and CTFd require the training facilitator to decide on the deployment strategy and execute it. SQLiTrainer has the most customizable exercises, while Hack The Box and Avatao can customize the set of exercises a training group will receive. CTFd can be customized based on the teacher needs.

IV. Requirements

Based on the previous analysis and the authors previous experience with eLearning tools in computer engineering education, for a system to be suitable for academic uses, a recommendation can be made as a set of requirements that need to be fulfilled. Target audience of the system should be students with no previous knowledge of SQLi or other software vulnerabilities. This requires a system with exercises focused only on SQLi. Exercises combining SQLi with other vulnerabilities are out of scope, as the target audience should have knowledge of multiple vulnerabilities and understand how they can be exploited together. Further requirements can be defined using the Cyber Taxi, making it suitable for comparison with the existing tools.

		Juice Shop	Hack The Box	Avatao	CTFd	picoCTF	SQLiTrainer
Techical Setup	Environment structure	Hosting. Each user has a dedicated instance of the target application.	Commercial E-Learning and Collaboration platform	Commercial E-Learning platform	Commercial E-Learning platform	E-Learning platform	Hosting. Each user has a dedicated instance of the target application.
	Deployment	On-premise and Cloud			Cloud		On-premise and Cloud
	Orchestration	Using MultiJuicer [x] exercises can be fully automated	Full automation	Full automation	Full automation	Full automation	Full automation
Audience	Sector	Academic and Private	Academic, Private and Public	Academic and Private	Academic, Private and Public	Academic	Academic and Private
	Purpose	Raise awareness and increase skill level	Raise awareness and increase skill level	Raise awareness and increase skill level	Raise awareness and increase skill level	Raise awareness and increase skill level	Raise awareness and increase skill level
	Proficiency level	Beginner to expert	Beginner to expert	Beginner to expert	Beginner to expert	Beginner to expert	Beginner
	Target audience	Students and IT Professionals	Students, IT Professionals and IT Specialists	Students and IT Professionals	Students, IT Professionals and IT Specialists	Students	Students
Training Environment	Training Type	Jeopardy style Capture The Flag (CTF)	Attack-only and attack-defense CTF, Cyber Training Range	CTF	CTF	CTF	CTF
	Scenario	Problem-driven, not supervised	Problem- and storyline-driven, not supervised	Problem- and storyline-driven, not supervised	Problem-driven, not supervised	Problem-driven, not supervised	Problem-driven, not supervised
Training Setup	Scoring	Awarding. Solved challenges are automatically detected.	Awarding. User is awarded with points, ranks and badges.	Awarding	Awarding. Solved challenges are automatically detected.	Awarding. Solved challenges are automatically detected.	Manual assessment
	Roles	No specific roles	Red and Blue teams when done in team setup	No specific roles	No specific roles	No specific roles	No specific roles
	Training Mode	Single	Single or Team	Single	Single or Team	Single	Single
	Customization Level	Specific. User interface and initial database state can be customized. Challenges cannot be customized.	Specific. Training can be customized using Dedicated Labs where administrators can pick set of exercises for training group.	Specific. Administrators can pick set of exercises for training group.	Specific. User interface and initial database and target database can be customized.	None	Specific. User interface, initial database state and target database can be customized.

Fig. 7. Clasification of the selected tools

 ISBN 978-86-7466-930-3

Audience
- Target audience: Students
- Sector: Academic
- Proficiency level: Beginner
- Purpose: Raise awareness and increase skill level

Training environment
- Training Type: Capture The Flag
- Scenario:
    - Non supervised
    - Problem- or Storyline- driven
    - Target of the challenge is application solved by exploiting SQLi vulnerability

Training setup
- Scoring: Assessment, as results of the participants should be comparable
- Roles: No specific roles
- Training mode: Team, or Individual during assessments
- Customization Level: Specific, as it should be customized for each training group

Technical setup
- Environment structure: Online platform in the sense of E-Learning platform, or hosting
- Deployment: Both On-premise and Cloud
- Orchestration: Full degree of automation with modular design of the application

Capture The Flag is the most practical training type for students. Students can be split in groups, each group having the same flag. As flags are known upfront, they can be checked automatically once the user submits them. Depending on modularity of the application, additional customization may go along, like each group having a specific user interface, initial database state, and SQL database, like PostgreSQL or MySQL.

For the selected tools to be fully compliant with the requirements, the following features should be introduced:

- Juice Shop: Customization of the exercises, providing different flags per training group
- Hack The Box: Exercises focused only on SQLi
- Avatao: More exercises focused on SQLi or customization of the existing ones
- picoCTF: introduce a specific SQLi category

As it can be seen additional effort is needed in order to use existing platforms in academia since the two very important requirements, i.e. exercises focused only on SQLi and customization of exercises cannot be expected to be fulfilled. On the other hand dedicated tools, like SQLiTrainer and frameworks, like CTFd, will fulfill all of the requirements, but will require even more effort to be created.

V. Example Of Quantitative Analysis

In this section we will give an example of a quantitative analysis using the results from the table represented in Fig. 7. and the requirements of the previous chapter. Each category in the previous mentioned table (technical setup, audience, training environment and training setup) will be graded on the scale from 1 to 5 based on how much the tools and the CTFd framework satisfy the requirements from the previous chapter.

Having that in mind we can draw a conclusion on how much each subcategory in each category weights in the final score of that category. Because all of the analyzed tools and CTFd framework satisfy the requirements of categories audience and training environment, on these two categories they all get the grade of 5.

That leaves us with two categories left to be graded, and those are training setup and technical setup. Here we can define how much each subcategory has weight in the final grade of its category based on the values for those subcategories from the previous chapter and from the table represented in Fig. 7. These specific weights are defined based on authors personal experiences with laboratory exercises on the Advanced Network and System Security course at the University of Belgrade, School of Electrical Engineering.

Training setup
- Scoring
    - Automatic assessment weights 40%
    - Manual assessment weights 20%
- Roles – 0% of the category grade
- Training mode
    - Single and team mode weights 20%
    - Only single or team mode weights 10%
- Customization Level
    - None weights 0% of the category grade
    - Ability to only pick a set of exercises for group weights 10%
    - Interface and initial database can be customized weights 20%
    - Initial database state and target database can be customized weights 30%
    - Initial database state, target database, and challenges can be customized weights 40%

Technical setup
- Environment structure
    - Noncommercial platform weights 50%
- Deployment
    - On premise and cloud weights 20%
    - Only on premise or on cloud weights 10%
- Orchestration
    - Full automation weights 30%

Based on these assessments we can see the final grade of each category and the average grade of each analyzed tool in a table represented in Fig. 8.

 ISBN 978-86-7466-930-3

	Juice Shop	Hack The Box	Avatao	CTFd	picoCTF	SQLiTrainer
Techical Setup	4	2	2	2	4	5
Audience	5	5	5	5	5	5
Training Environment	5	5	5	5	5	5
Training Setup	4	4	3	5	3	3
Average	4.5	4	3.75	4.25	4.25	4.5

Fig. 8. Grades of the selected tools

VI. Conclusion

This paper presented the need for having tools to learn SQLi vulnerability in an interactive way and listed the existing tools. Selected tools were categorized based on the parts of the Cyber Taxi. The same taxonomy was used to compare the selected tools, leading to suggestion of requirements that a tool to be used in academia needs to fulfill. Conclusion is that effort needed to adapt existing tools to fulfill these requirements sometimes can exceed the effort needed to create a dedicated tool from scratch and sometimes it is even not possible to customize existing tool. However, once the requirements are clear it is always useful to check if there are existing tools that can fulfill them before starting to create a new tool. The quantitative analysis that was conducted in this paper showed how once the requirements are defined it is possible to quantify them and create a unique benchmark for each individual teacher and each individual course.

References

[1] IEEE Computer Society and ACM, Curriculum Guidelines for Undergraduate Degree Programs in Software Engineering, Available at: http://www.acm.org/binaries/content/assets/education/se2014.pdf (Accessed September 2021)
[2] Đ. Madić, Ž. Stanisavljević, SQLITRAINER - Sistem za učenje o SQLi sigurnosnim propustima u aplikacijama, ETRAN 2021, RT1.2, September 2021
[3] Knüpfer, M., Bierwirth, T., Stiemert, L., Schopp, M., Seeber, S., Pöhn, D. and Hillmann, P., 2020, September. Cyber Taxi: A Taxonomy of Interactive Cyber Training and Education Systems. In International Workshop on Model-Driven Simulation and Training Environments for Cybersecurity (pp. 3-21). Springer, Cham.
[4] Juice Shop, Available at: https://owasp.org/www-project-juice-shop/ (Accessed September 2021)
[5] OWASP, Available at: https://owasp.org/ (Accessed September 2021)
[6] E-Commerce news, Available at: https://ecommercenews.eu/800000-online-stores-europe/ (Accessed September 2021)
[7] Juice Shop code repository, Available at: https://github.com/bkimminich/juice-shop (Accessed September 2021)
[8] Hack The Box, Available at: https://www.hackthebox.eu/ (Accessed September 2021)
[9] Avatao, Available at: https://avatao.com (Accessed September 2021)
[10] CTFd, Available at: https://ctfd.io/ (Accessed March 2022)
[11] PicoCTF Available at: https://picoctf.org/ (Accessed March 2022)

 ISBN 978-86-7466-930-3

Automated grading system for picoComputer assembly codes integrated within E-Learning platform

Jovan Đukić, Vladimir Jocović, Marko Mišić, *Member, IEEE,* and Milo Tomašević

***Abstract*—Obtaining programming skills is one of the most important prerequisites for a future career of every electrical or software engineer. The programming expertise is best acquired by gradually advancing from simpler to more complex programming paradigms, architectures, and languages. That being the case, a restrictive educational computer architecture – picoComputer, along with a development environment, was developed at the University of Belgrade, School of Electrical Engineering to early expose the students to the concepts of assembly language programming. Having in mind that programming skills are successfully attained only through practical work, such as homework assignments, projects, and laboratory exercises, some more contemporary picoComputer simulation environments were implemented, including MessyLab desktop application and a Picosim web-based solution. However, programming courses at our school are massive and require the utilization of online learning platforms, aiming to properly achieve a scalable learning process. Hence, we employed Moodle E-Learning platform, as well as the CodeRunner plugin, to facilitate and accelerate the teaching and assessing processes in both of our major programming courses. CodeRunner plugin supports various widespread programming languages and is also highly programmable, which is why the integration of picoComputer architecture within a contemporary learning system arose as an opportunity.**

***Index Terms*—E-Learning; automated code assessment; Moodle; picoComputer**

I. Introduction

Strong programming expertise is one of the fundamental abilities of a contemporary software engineer and a necessary quality of an electrical engineer, as well. Gaining programming practice is essential for not only solving real-world problems in software but also for memory sharpening and achieving an ability to efficiently resolve various problems that are seemingly outside of the programming scope.

Jovan Đukić is with the School of Electrical Engineering, University of Belgrade, 73 Bulevar kralja Aleksandra, 11020 Belgrade, Serbia (e-mail: dj@etf.bg.ac.rs)

Vladimir Jocović is with the School of Electrical Engineering, University of Belgrade, 73 Bulevar kralja Aleksandra, 11020 Belgrade, Serbia (e-mail: jocke@etf.bg.ac.rs)

Marko Mišić is with the School of Electrical Engineering, University of Belgrade, 73 Bulevar kralja Aleksandra, 11020 Belgrade, Serbia (e-mail: marko.misic@etf.bg.ac.rs), (https://orcid.org/0000-0002-7369-4010).

Milo Tomašević is with the School of Electrical Engineering, University of Belgrade, 73 Bulevar kralja Aleksandra, 11020 Belgrade, Serbia (e-mail: mvt@etf.bg.ac.rs)

An effective approach to adopting good programming skills requires a strong theoretical foundation, which is being acquired through traditional methods of lecturing, in addition to a practical approach, which includes homework assignments, larger projects, and laboratory exercises [1].

Programming courses at the School of Electrical Engineering, the University of Belgrade, are mandatory for all first-year students and they are organized into two one-semester courses - Programming 1 and Programming 2. Both courses are mainly focused on studying programming languages (Python and C) and introduce different programming paradigms. They start with a low-level assembly language and continue with a more complex procedural and to some extent object-oriented programming paradigm. Moreover, course topics are permeated with basic data structures and code complexity topics.

Our first-year programming courses are attended by a vast number of students (up to a thousand). These massive courses impose a lot of overhead regarding the process of qualitative assessing the students' work and administering the course contents, as well. Hence, there was a need to establish online learning platforms and other tools intending to ease the whole process of managing these huge courses, as well as disburdening the already overexerted teaching staff. Our first experiences with Moodle E-Learning platform in programming courses are described in [2]. In our previous efforts, we also had a good experience with Moodle e-learning platform for other computer engineering courses [3], as well as with other tools for student assessment [4], analysis of results [1], and source code plagiarism detection [5, 6] which are all widely used in programming education. For all those reasons, we decided to implement appropriate support for the emulation of picoComputer assembly codes in Moodle e-learning platform, as well. We describe our motivation and the details of implementation in the rest of the paper.

The second section expresses the reasons behind the choice to move mandatory programming courses to the e-learning Moodle platform and the course organization within it, as well as the examination process using CodeRunner plugin. The third section presents in more detail the in-house developed architecture for teaching assembly language programming - picoComputer (pC). The fourth section describes the present-day implemented system for compilation, execution, and evaluation of source codes written in the pC assembly language and its integration with the CodeRunner plugin within Moodle platform. The fifth section illustrates the evaluation process and the results obtained by system testing. Finally, a brief conclusion and future work are given in the last section.

 ISBN 978-86-7466-930-3

II. MOODLE PLATFORM AND CODERUNNER PLUGIN IN PROGRAMMING COURSES

The programming courses lectures at the School of Electrical Engineering are held by professors and teaching associates. The professors mostly teach the theoretical aspects of the currently studied programming topics, giving emphasis to some important essences needed for successful mastering of the practical programming tasks. Bearing that in mind, the teaching associates organize auditory exercises in a more practical manner, thus those classes are dedicated to programming practices only.

Each elementary problem and some intermediate programming tasks are conducted alongside students, aiming to strengthen and solidify their ascending programming skills, as well as to introduce new approaches to solving programming problems. Simple programming tasks were solved using an integrated development environment, such as PyCharm for Python or Microsoft Visual Studio for C, while the more complex ones were worked out on the Moodle platform using the CodeRunner plugin.

Moodle is an online learning platform that allows teachers to create courses for students and to grade their work in those courses using tests. The platform supports a variety of plugins, and we found the CodeRunner plugin the most useful for our grading purposes. This plugin introduces a new type of question, which allows teachers to assess and grade students' source codes. Correctness of the students' codes is partially verified using an automated testing process implemented by the teaching associates, who managed to appropriately configure the plugin using a custom Python script. This feature places the CodeRunner plugin at the top of the list of supported plugins. Unfortunately, some code characteristics still need to be checked manually, e.g., coding style and efficiency.

The example that demonstrates the usage of the Moodle platform is shown in Figure 1. It illustrates the exercise concerning the linked list data structure. The exercise is carried out in C programming language and consists of basic operations performed on linked lists: insertion and deletion of elements, list traverse operations, etc. Before a problem is approached practically, the topic is explained using a PowerPoint presentation. The task itself is straightforward and performed on a simple linked list of integers.

Intending to make the topic of linked lists more interesting, a big task is organized as a series of smaller tasks for a more comprehensive understanding. The task is divided into smaller task sets of varying difficulty. Former task sets consist of commonly used linked list operations and latter task sets functionally depend on the previous task sets. This way the goal is to incrementally build and test the solution and to teach students one of the most important programming principles – code reusability.

As shown in Figure 1, a question has a small table at the top of the page which contains the test input data and the expected output data. Below the test cases table, there is a text area for the code itself. The CodeRunner plugin also includes syntax coloring which is an additional advantage for the students since it can indicate the errors that would be very hard to find in a classical exam notebook.

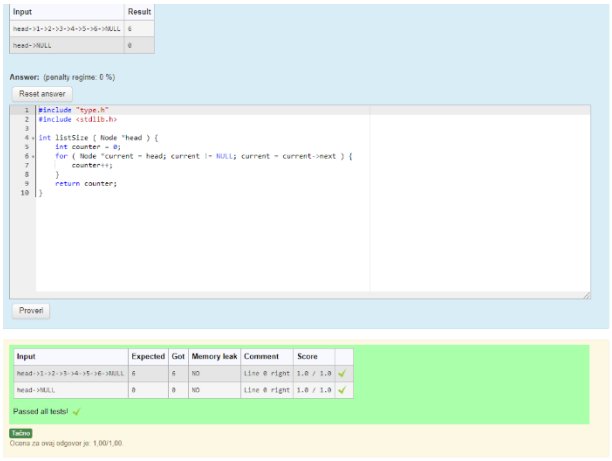


Figure 1 A CodeRunner question in C

After the required piece of code is written in the provided text area, the students can check its correctness by clicking on the check button. When the button is clicked, the contents of the text area are sent to a server dedicated to checking and grading CodeRunner questions. The server first compiles the given code and executes it using the supplied test cases. After execution, the server collects the output and compares it with the expected output.

The main advantage of the CodeRunner plugin for Moodle platform is that the entire process of code checking is configurable [7]. This is achieved through Python scripts which are executed each time a student checks the question. Furthermore, the comparison of the expected and collected output can be graded line by line, thus allowing the students to receive partial points for each successfully passed test case. After the outputs are compared, the result table, shown below the text area in Figure 1, is created giving feedback and their scores to the students, while pointing them to possible errors in the code.

In the last few years, teaching associates were able to master the craft of the plugin configuration and managed to successfully port programming tasks written in Python and C programming languages to the CodeRunner plugin. These high-level programming languages are the foundation of our mandatory first-year courses. Details of the porting process of these high-level languages and further information are presented in [2]. The CodeRunner plugin was successfully used at other universities in other contexts for Python and C++, as well [8].

However, before introducing the students to the concepts of high-level programming languages, they are taught some elementary concepts of low-level assembly programming. Being a relatively minor part of the course, it required an underlying educational architecture that would be quite restrictive, and that's why the aforementioned picoComputer architecture is envisioned and developed. Considering the nature of the low-level languages, the written assembly code can frequently be hard to read, maintain and debug. Even when the source code is syntactically correct, its possible semantic flaws could be tedious to discover. Until recently, teaching associates had to manually inspect the source code, which can devour plenty of time and energy, even for simple programming problems.

Taking these circumstances into account and having a positive experience gained from porting high-level

 ISBN 978-86-7466-930-3

programming languages such as Python and C to Moodle using the CodeRunner plugin, teaching associates decided to establish a system that could verify and test the students' solutions written in the picoComputer assembly language on the e-learning platform. Nevertheless, the task of implementing such a system is inherently harder than above mentioned porting challenges. Teaching associates had to provide not only a configured environment for executing students' source codes using test examples but also an implementation of a compiler and an emulator for a source code written in assembly language was necessary. These components were not needed in previous porting undertakings, since there are numerous compilers and interpreters for widespread programming languages, including C and Python.

III. PICOCOMPUTER

In 1989. Prof. Jozo Dujmović designed a computer architecture named picoComputer (pC) [9] and developed a DOS application *pC Assembler and Simulator* (pCAS). His intention was to facilitate the teaching and understanding of the assembly languages, which are naturally, due to their low-level nature, to some degree demanding. Since only a part of the introductory programming course is devoted to low-level programming, the pC is designed as a quite restrictive architecture as implied by its name. However, even with its restrictive scope, pC is still very useful. In order to provide more convenient environments, two tools have been recently developed at our school, MessyLab IDE [10], and a web online environment called picoSim [11].

Although the picoComputer architecture is more than 30 years old, it is still relevant nowadays. Its aim is to provide a framework for demonstrating assembly-level programming. It follows the classical Von Neumann architecture and, even though the characteristics of various components have changed throughout time and the instruction sets are getting more and more complex, the basic principles of computer structure have not changed a lot. The picoComputer generally consists of the Central Processing Unit, Random Access Memory, and Input/Output devices, which are all connected using a shared Bus, as shown in Figure 2.

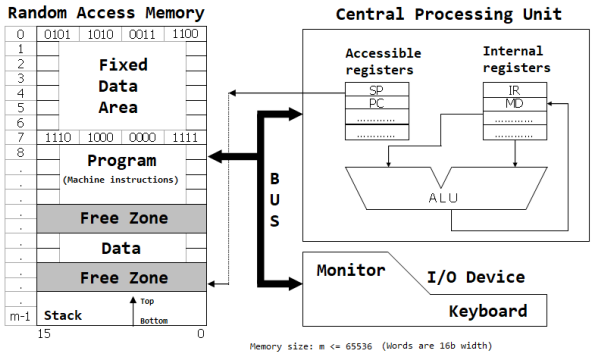


Figure 2 The picoComputer architecture

The Central Processing Unit has several internal registers, yet none of them are directly accessible as the instruction operands. Because of a limited instruction format, there are no general-purpose registers. Still, Program Counter (PC), which points to the next instruction to be executed, and Stack Pointer (SP), which indicates the top of the stack, are registers that can be indirectly manipulated by certain instructions. The value of the PC register is either incremented after an executed instruction or can be directly loaded with the branch address by a control instruction. SP register value is affected by subroutine handling instructions and.

The Random Access Memory consists of 65536 locations (memory words), which means that memory addresses are 16 bits wide, while each location is, also, 16 bits wide. The memory is logically divided into two sections: Fixed Data Area and Free Area. Fixed Data Area includes the first 8 locations, which are directly accessible through direct memory addressing. Free Zone is comprised of the remaining locations and these locations are only accessible indirectly, through another location from the Fixed Data Area, using memory indirect addressing. These locations can be used arbitrarily. The third addressing mode is the immediate addressing where the operand is found in the instruction itself.

The input device is a keyboard, and the output device is the monitor. Numerical data can be entered using a keyboard, while the screen presents the contents of certain memory locations. The input/output operations are blocking operations. Consequently, there is no need for polling a status register. However, parallelism is not supported.

Every picoComputer program consists of two sections: the directive section, and the instructions section. There are two kinds of directives: symbol definition directives and the origin directive. A symbol definition directive is used to assign numerical values to symbols to improve code readability. These symbols are replaced by their numerical values in the assembling process. Labels are an implicit means of symbol definitions, and they can be specified by an identifier attached to any instruction. The origin directive defines the starting memory location where the executable code (instruction section containing instructions following the origin directive) resides.

Every instruction is defined by its symbolic mnemonic and a variable-length comma-separated list of operands. The picoComputer format provides up to 3 operands in an arbitrary instruction. Instructions can occupy one or two memory words (16 or 32 bits). The operation code and three operand fields are encoded in four 4-bit nibbles of the first word, as shown in Figure 3.

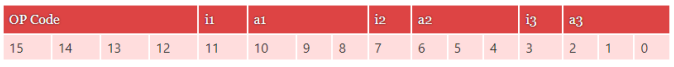


Figure 3 Typical picoComputer one-word instruction format

Hence, a maximum of 16 different operation codes are supported. Each operand specification consists of *i* field (1 bit) and *a* field (3 bits), where *i* indicates the memory addressing mode (0 for direct or 1 for indirect), while *a* field represents an address from Fixed Zone Area (0-7). The 16-bit immediate operand, when supplied, is stored in the second instruction word. The pC instruction set consists of the integer arithmetic instructions (addition, subtraction, multiplication, division), data transfer instruction (scalar or

 ISBN 978-86-7466-930-3

vector move), conditional branches, subroutine call and return, I/O instructions, and stop instruction.

IV. ASSESSMENT OF STUDENTS' PICOCOMPUTER ASSEMBLY PROGRAMS

As stated, in the second chapter, CodeRunner allows its users to provide a custom Python script for the purpose of grading and assessing. This script is executed in a preconfigured CodeRunner plugin environment, which provides numerous predefined variables. These variables can be used to obtain a variety of information about students from Moodle (i.e., student profile information) and, more importantly, the answer submitted by the student through the CodeRunner form. Given that the pC is our custom assembly language and that the answer is given in a text form, the authors had to build a custom compiler and an emulator to be able to grade and assess students' answers.

The compilation phase consists of text manipulation and performs syntax and semantic checks specified by the rules of the assembly language. During this stage, the text is split into individual lines, which are checked separately. If the process is successful, the result is a Python list data structure of integers, where each element represents an individual memory location. However, if there is an error in the code, the result of the compilation phase is a list containing descriptions of each individual erroneous line (line number and the error description). The unsuccessful compilation phase is depicted in Figure 4.

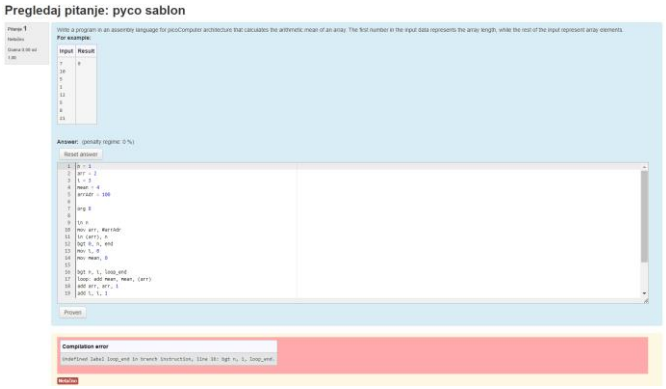


Figure 4 Unsuccessful pC compilation phase

If the compilation phase is successful, the following step is the emulation phase. The input data for the emulation phase is also a list of integers (i.e., memory locations), the address of the first instruction provided by the origin directive and a list of integers representing the input data. The emulation phase is a simple *for*-loop, which reads instructions one by one, executes them and stores their results in the memory. The exceptions to this workflow are the IN and OUT instructions. The IN instruction reads one or more numbers from the input data list, while the OUT instruction writes the content of one or more memory locations to the output list. This output list is later used for comparison with the expected results. Runtime checks are also performed during this stage. Given the restricted nature of the pC and the fact that only integer data type is supported, the only runtime check performed is the division-by-zero check. This is depicted in Figure 5.

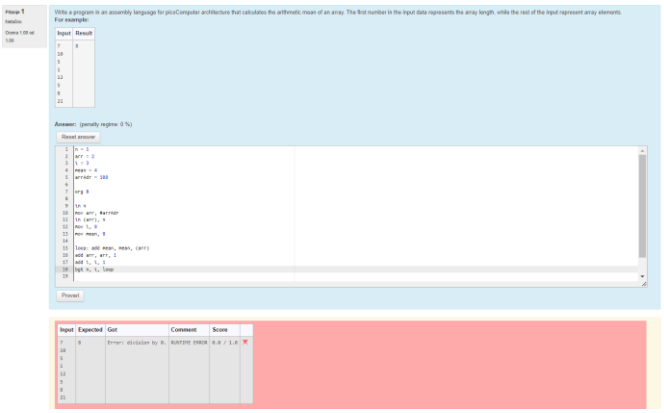


Figure 5 Runtime error checks performed

If both the compilation and the emulation phases are successful, the result of the emulation phase is compared to the expected result and the final grade is formed. The comparison is performed on a line-by-line basis, where each line gives the same number of points. Finally, the student is presented with the score table as shown in Figure 6.

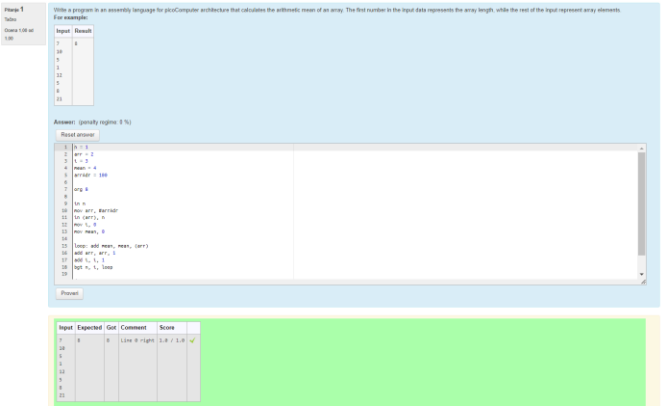


Figure 6 Score table for a program

V. SYSTEM TESTING AND EVALUATION PROCESS

During the development, the system was tested with custom-written programs, which cover all possible valid and invalid instruction formats. Valid programs consist of instructions as specified by the pC instruction format. These programs are relatively easy to write since there is a limited number of possibilities that are in accordance with the syntax and semantic rules. However, the number of invalid formats is far greater in number and, therefore, it is impossible to cover all of them. Hence, the system was also tested with the source codes of the students in the previous years.

Until recently, during exams students used the MessyLab desktop application to write and self-check their solutions. Regardless, the students used the Moodle platform to submit their answers, which were graded manually. We used these answers to perform evaluation by comparing manual scores given by the teaching associates and the scores given by the system. The system was evaluated on students' programs after the exams have passed and the results of both approaches were quite similar. We sincerely hope that the live results will be as good as these.

 ISBN 978-86-7466-930-3

VI. CONCLUSION

In this paper, we have presented the migration process of a restrictive educational picoComputer architecture to the online e-learning Moodle platform using a dedicated CodeRunner plugin for our introductory programming course. With this new feature, we have successfully fully moved all our programming activities within the Programming 1 course to the LMS. Moodle and CodeRunner are extensively used during auditory exercises in the computer lab and examination. The students reacted very positively to this innovative activity.

Having migrated all programming languages in our mandatory programming courses, it may seem that the future work lacks required matter to be considered worthy. However, the is a significant potential concerning the available possibilities to parametrize and configure these types of systems not only to achieve more sophisticated means of grading, yet also to broaden the spectrum of conceivable programming task types. In the future, we have an intention to develop various learning activities in Moodle, as well to extend our coding exercises pool with more assignments written in different programming languages. Moreover, we have in mind to migrate the rest of our programming courses to such type of learning and examining.

ACKNOWLEDGMENT

This work was supported by the Science Fund of the Republic of Serbia, grant no. 6526093, AI-AVANTES, as well as the Ministry of Education, Science, and Technological Development of the Republic of Serbia (III44009 and TR32047). The authors gratefully acknowledge the financial support.

REFERENCES

[1] M. Mišić, M. Lazić, and J. Protić, "A software tool that helps teachers in handling, processing and understanding the results of massive exams," in *Proceedings of the Fifth Balkan Conference in Informatics*, 2012: ACM, pp. 259-262.

[2] V. Jocović, J. Đukić, and M. Mišić, "First Experiences with Moodle and Coderunner Platforms in Programming Course," in *Proceedings of the Tenth International Conference on e-Learning*, Belgrade Metropolitan University, Belgrade, 2019, pp. 81-86.

[3] D. Drašković, M. Mišić, and Ž. Stanisavljević, "Transition from traditional to LMS supported examining: A case study in computer engineering," *Computer Applications in Engineering Education*, 2016.

[4] A. Bošnjaković, J. Protić, D. Bojić, and I. Tartalja, "Automating the Knowledge Assessment Workflow for Large Student Groups: A Development Experience," *International Journal of Engineering Education*, vol. 31, no. 4, pp. 1058-1070, 2015 2015.

[5] M. Mišić, Ž. Šuštran, and J. Protić, "A Comparison of Software Tools for Plagiarism Detection in Programming Assignments," *International Journal of Engineering Education*, Article vol. 32, no. 2, pp. 738-748, 2016 2016.

[6] M. J. Mišić, J. Ž. Protić, and M. V. Tomašević, "Improving source code plagiarism detection: Lessons learned," in 2017 *25th Telecommunication Forum (TELFOR)*, 2017: IEEE, pp. 1-8.

[7] R. Lobb and J. Harlow, "Coderunner: A tool for assessing computer programming skills," *ACM Inroads*, vol. 7, no. 1, pp. 47-51, 2016.

[8] D. Croft and M. England, "Computing with CodeRunner at Coventry University: Automated summative assessment of Python and C++ code," in *Proceedings of the 4th Conference on Computing Education Practice 2020*, 2020, pp. 1-4.

[9] J. J. Dujmović, Programski jezici i metode programiranja – odabrana poglavlja. *Akademska misao*, 2004.

[10] M. Anđelković. "MessyLab project." http://messylab.com/ (accessed April 2022).

[11] N. Miljković. "picoSim project." https://picosim.app/ (accessed April, 2022).

Pandemic Support System Modelling and Implementation as Integral Part of Computer Science Courses

Nenad Petrović

***Abstract* — In this paper, exercises related to modelling and implementation of pandemic-tackling systems are proposed as integral part of computer science university courses. The presented case study considers the perspective of two bachelor degree courses taught at the third year of Computer Science and Informatics track at University of Niš, Faculty of Electronic Engineering in Serbia covering both hardware and software design: Microcomputer Systems and Information Systems. For the first course, an indoor safety system based on Intel 8086 and additional components (8251, 8255 and 8259) is presented. On the other side, the main topic of the second one is Java Enterprise Edition (JEE)-based information system development, while the presented example shows application providing efficient pandemic-related data management (coronavirus tests and vaccination).**

***Index Terms* — COVID-19; coronavirus; education; Intel 8086; Java Enterprise Edition (JEE).**

I. Introduction

The ongoing coronavirus pandemic has brought numerous challenges not only to healthcare and medical science-related personnel, but when it comes to engineering and information sciences professionals as well. Efficient pandemic-related data management in synergy with leveraging the collected information are recognized as key factors when it comes to role of information systems in global battle against COVID-19 [1]. On the other side, IoT and embedded devices are one of crucial means for disease spread reduction (such as mask detection and contactless temperature measurement) [2]. In this paper, we introduce two exercises aiming Information Systems and IoT bachelor degree courses at Faculty of Electronic Engineering, University of Niš in Serbia. The first one covers the design and development of software leveraging pandemic data, while another is a control unit of indoor safety system. The goal is to show how students can become aware of their role as computer engineers among other professionals tackling COVID-19, while studying Java Enterprise Edition software development and programming microcontrollers, on the other side.

The exercise case studies presented in this paper are inspired by author's intensive work in area of pandemic support information technology solutions since the beginning of COVID-19 outbreak. In [3], an affordable IoT-based indoor safety system for mask check and temperature detection leveraging Raspberry Pi and Arduino Uno was introduced. Moreover, the works from [4, 5] show methodology for efficient pandemic resource planning, such as vaccine allocation and tests. Furthermore [6] shows air quality regulation system for coronavirus spread reduction, while [2] addresses green certificates checking relying on blockchain. Finally, [7] shows adoption of data mining techniques in area of post-COVID tourism. On the other side, author also introduced exercises related to coronavirus protection within Logic Design [8, 9] course and microcontroller-related part of Microcomputer Systems [9, 10].

Nenad N. Petrović is with University of Niš, Faculty of Electronic Engineering, Aleksandra Medvedeva 14, 18000 Niš, Serbia, (e-mail: nenad.petrovic@elfak.ni.ac.rs), (https://orcid.org/0000-0003-2264-7369)

II. Background

A. Information Systems Course

Information Systems is third year bachelor degree course, mainly covering Java Enterprise Edition (JEE) [11] and related technologies within practical sessions. The implementation-related assignment is homework project scored by 25% of the overall coursework. The underlying architecture is depicted in Fig. 1. In such application, user provides the corresponding input data and selects desired options via web browser-based interface. Moreover, in order to generate the expected results, business logic layer is responsible for calculations and data manipulation. Additionally, at some point it might rely on external microservices deployed within containers for some of the tasks. During this processing within business layer, objects holding the relevant data about events, users and other entities might be persisted within database or retrieved from there Finally, the outcome is shown back to the end-user inside web page. Later, the acquired data can be further leveraged for various types of predictions relevant to business goals.

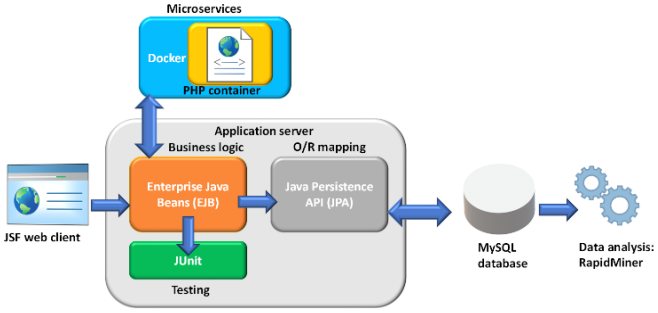


Fig. 1. Information Systems course project architecture.

An overview of components corresponding to the topics covered by the practice-oriented part of Information Systems course is given within Table 1.

 ISBN 978-86-7466-930-3

TABLE I
INFORMATION SYSTEMS TOPIC OVERVIEW

Component	Type	Role
Java Server Faces (JSF) [11]	Java-based framework for web applications (front-end)	User data input and result visualization
Enterprise Java Beans (EJB) [11]	Business logic layer components for application JEE back-end	Calculations and data manipulation
JUnit [12]	Testing framework	Unit testing of business layer functionalities based on assert checking
Java Persistence API (JPA) [11]	Object-relational (O/R) mapping specification	Persistence of entities (Java class objects) in form of database rows
MySQL [13]	Relational database	Storage of relevant data within tables
Docker [14]	Containerization engine	Microservice implementation and deployment within isolated environment
RapidMiner [15]	Automated machine learning tool	Predictions using classification, regression, clustering and association rules

B. *Microcomputer Systems Course*

Microcomputer Systems are also obligatory third year bachelor degree course, but focused on hardware design instead. It consists of two main parts: 1) Intel 8086-based systems [16] 2) PIC16-family microcontrollers. In this paper, we focus on the first part, which represents around 30% of the overall course work, split into lab session exercises and written exam. When it comes to that topic, several additional components are also covered, as shown in Table 2.

TABLE II
MICROCOMPUTER SYSTEMS TOPIC OVERVIEW

Component	Type	Role
8086	16-bit microprocessor	Control unit and data processing
8255	Parallel data peripheral interface	Output to devices such as 7-segment displays and LEDs or input handling, such as button press
8251	Serial data transfer interface	Receive/transmit data via single line from serial devices and convert it to parallel in order to be usable by 8086
8259	Programmable Interrupt Controller	Detecting changes on interrupt lines and providing mechanism for response to events which occurred in the external environment

III. CASE STUDIES

A. *COVID-19 Healthcare System for Test and Vaccination*

The goal of the project assignment is to model and implement a healthcare information system aiming to support COVID-19 testing and vaccination processes. The complete source course of the project is given within public online GiHub repository: https://github.com/penenadpi/is2022_covid.

It gives the ability to enter the related data using corresponding JSF pages, as shown within screenshot of test-related interface (Fig. 2). On the other side, it enables the storage and retrieval of relevant information about vaccination as well, such as ambulance identifier, date, vaccine type, the number of dose and person identifier.

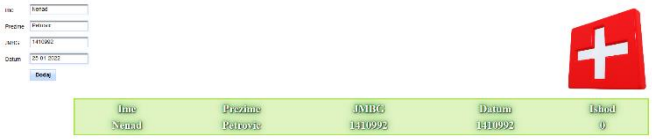


Fig. 2. JSF page for COVID-19 test records.

We can distinguish between three main types of Java class components within the application (as illustrated in Fig. 3): 1) entity classes – responsible for data persistence about relevant objects (*VaccinationRecord* and *TestRecord*); 2) session beans services – implementation of business logic services together with their corresponding interfaces (*VaccinationService* and *TestService*); 3) controllers - functionalities related to a single JSF page (*VaccinationController* and *TestController*). The underlying flow can be described as follows: controllers are triggered by clicking on buttons within the JSF page, while inside them, the calls to corresponding services leveraging the data (storing or reading) and performing processing are invoked. These operations are executed against entity objects which are permanently persistent within database relying on O/R mapping.

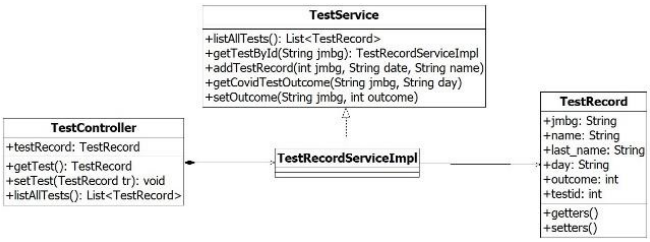


Fig. 3. Excerpt of UML class diagram for coronavirus test records.

On the other side, when it comes to testing, JUnit framework for unit testing in Java is covered by this course. Unit tests aim to check whether lowest level executable units (such as single method in object-oriented program or function) give correct result. For that purpose, test cases covering relevant features of the project are written. An example which checks whether new vaccination records are inserted correctly is given in Fig. 3. The test case consists of three main parts: 1) pre-condition: assert which needs to hold before execution of the test itself 2) test steps: the functional call to the method being tested 3) post-condition: condition which should be true after the execution of test steps. As it can be seen in Fig. 4, the precondition checks

 ISBN 978-86-7466-930-3

whether vaccination record for given person does not exist. Then, the test itself adds vaccination record. Finally, the post-condition checks if insertion had been successful by retrieving the vaccination record of provided person and testing if it is not null value.

```
@Before
public void testPrecondition()
{
    VaccinationRecord vr1=service.getVaccByPersonId(14103);
    assertNull(vr1);

}

@Test
public void testCase1()
{
    service.createVaccinationRecord(14103, 2, "Sinopharm", "13-06-2021");

}

@After
public void testPostcondition()
{
    VaccinationRecord vr1=service.getVaccByPersonId(14103);
    assertNotNull(vr1);
}
```

Fig. 4. JUnit test case implementation.

When it comes to integration with microservices, the layer of business logic (*CovidTestService* session bean) interacts with PHP script deployed in a container via HTTP request. In Fig. 5, an example in context of our project assignment is shown. When it comes to insertion of COVID test results, the value of test outcome field is taken as result of PHP script deployed within web server inside Docker container. The script itself takes person identifier and current date as input, while the outcome is randomly generated (0 or 1), simulating the randomness regarding the probability of being infected.

```
public String getCovidTestOutcome(String pid, String day) throws IOException {
      String outcome="";
      URL obj = new URL("http://192.168.99.100/?pid="+pid+"&day="+day);
      HttpURLConnection con = (HttpURLConnection) obj.openConnection();
      con.setRequestMethod("GET");
      con.setRequestProperty("User-Agent", USER_AGENT);
      int responseCode = con.getResponseCode();
      System.out.println("GET Response Code :: " + responseCode);
      if (responseCode == HttpURLConnection.HTTP_OK) { // successful call
            BufferedReader in = new BufferedReader(new InputStreamReader(con.getInputStream()));
            String inputLine;
            StringBuffer response = new StringBuffer();

            while ((inputLine = in.readLine()) != null) {
                  response.append(inputLine);
            }
            in.close();
            outcome = response.toString();
            return outcome;
      } else {
            System.out.println("GET request not worked");
            return outcome;
      }

}
```

Fig. 5. Integration with Docker container microservice.

Finally, the last part of the project assignment involves leveraging the data collected by the described information system in order to make predictions which could be relevant to the pandemic support domain. For this purpose, we make use of RapidMiner tool, which gives intuitive and automated visual interface for machine learning. The presented example shows the adoption of regression, whose goal is to predict the value of numerical (continuous-valued) outcome. The layout of the underlying dataset based on the collected data is shown in Table 3. The goal is to predict vaccine demand (number of persons vaccinated) for given day. Relevant factors (input, independent variables) are day number, vaccine type, institution id and number of daily COVID-positive cases. On the other side, the predicted value (dependent variable) is the number of people who decided to take the vaccine.

TABLE III
VACCINE DEMAND PREDICTION DATASET HEADER

Day	Vaccine type	Institution	New cases	Demand

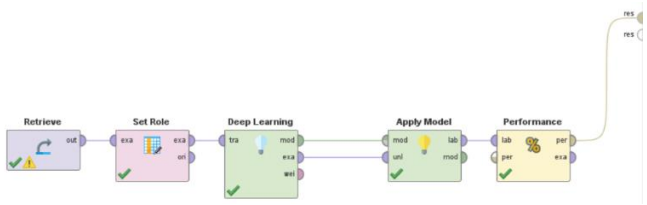


Fig. 6. Vaccination demand prediction workflow in RapidMiner.

The corresponding RapidMiner flow for regression-based prediction is shown in Fig. 6. The first step imports CSV containing all the data. After that, we select the target variable among the available columns. The third step is deep learning module which is trained to make predictions on 80% of the dataset. It contains 2 hidden layers, 30 nodes each and ReLU activation function, performs training in 10 epochs with adaptive learning rate. After that, the model is executed on test data and performance evaluated (mean relative error, which was around 4% in our case).

B. Control Unit of Coronavirus Indoor Safety System

On the other side, the goal of Microcomputer Systems assignment is design of control unit within COVID-19 indoor safety system (depicted in Fig. 7). Complete code with Proteus simulation is available on GitHub: https://github.com/penenadpi/8086covidsafety/blob/main/8086_8255_8259_8251_covid_safety.pdsprj

Each time new person arrives, two checks are performed by external devices: 1) temperature check – whether it is in normal range (less than 37°C) 2) mask check – if person wears mask or not. After that, the data is sent via 8251 component to the microprocessor, relying on 8259 for

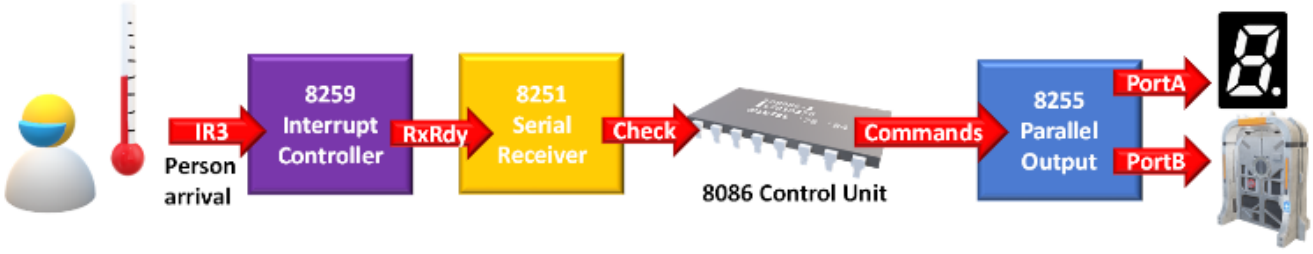


Fig. 7. 8086-based indoor coronavirus safety control system.

 ISBN 978-86-7466-930-3

interrupt handling (vector 171, line 3). In case that person passes both checks, then the door will be opened only in case that the current number of persons is in the allowed range for that room. Additional output peripheral devices are connected via ports of 8255: 1) green LED – turned on for opening door (PORTB, pin 1); 2) red LED – turned on for closing door (PORTB, pin 0); 3) 7-segment display (PORTA) – showing the current number of persons inside. However, when person leaves room, interrupt which decreases the number of persons inside the room is activated, as button on interrupt line 1, vector 169.

```
PROCED SEGMENT
ENTRANCE_CHECK PROC FAR
ASSUME CS:PROCED, DS:DATA

 MOV DX, CTRL
 IN AL, DX

 MOV DX, DAT
 IN AL, DX

 XOR AL, MASK_TEMP_CHECK
 JNZ CLOSE_DOOR

 CMP SI, 5
 JE CLOSE_DOOR

 INC SI

 MOV DX, PORTA
 MOV AL, NUMBERS[SI]
 OUT DX, AL

 MOV DX, PORTB
 MOV AL, GREEN
 OUT DX, AL

 JMP END

CLOSE_DOOR:
 MOV DX, PORTB
 MOV AL, RED
 OUT DX, AL

END:

 IRET
ENTRANCE_CHECK ENDP
PROCED ENDS
```

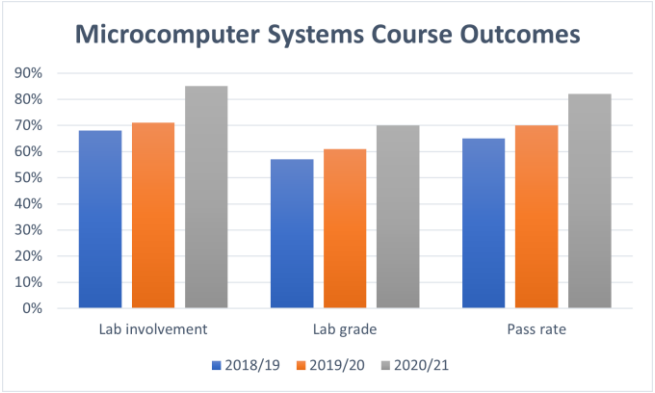
Fig. 8. 8086 interrupt procedure activated when 8251 receives data.

An excerpt of the solution containing the crucial part of the assembly code run on Intel 8086, related to interrupt triggered when new person arrives is shown in Fig. 8. As interrupt line activation means that 8251 is ready for receiving data, we set the corresponding data exchange address and receive the outcome of the entrance rules checks (mask and temperature) coming from other devices. After that, the received byte is compared against the only case which allows opening the door (MASK_TEMP_CHECK which is 18 hexadecimal). If it is not the case, the red LED will be turned on (door closed). Additionally, if person passes this check, it is also asked whether the current number of persons inside the room is below the maximum threshold (5 in this case). Only if it is true, the door will be opened (green LED turned on), the current number of persons incremented and shown on 7-segment display.

IV. Evaluation

The proposed approach of including pandemic-related exercises within computer science bachelor degree courses has been evaluated by observing the results before and after their adoption. The following aspects were considered: number of students involved into lab sessions, average lab exam grade and exam pass rate. Fig. 9a and 9b show graphical summary of results compared to previous academic years for Microcomputer and Information Systems courses, respectively.

When it comes to Microcomputer Systems, COVID-19 protection systems were also part of additional student challenges bringing bonus points which can change the whole written exam, where participation was around 15% of the total students involved.



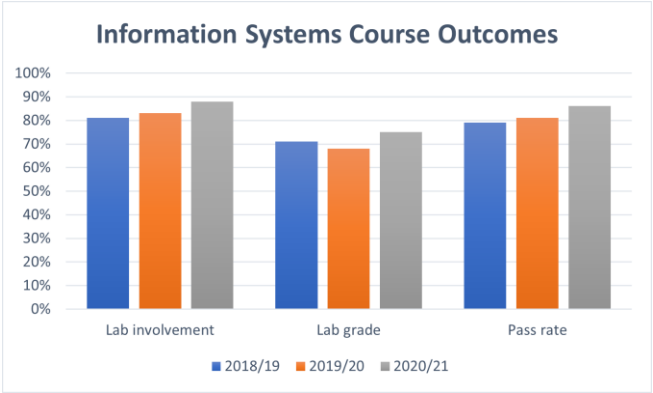


Fig. 9. Course outcomes before and after adoption of the proposed teaching methodology: a) Microcomputer Systems b) Information Systems.

V. Conclusion

In this paper, it was shown how the implementation of pandemic support solutions can be integrated within computer science university cases, considering the case studies of both hardware and software design. According to authors' observations, the adoption of such exercises in previous school year (2020/21) has shown several benefits for both of the considered courses: greater student involvement into lab sessions together with better lab

 ISBN 978-86-7466-930-3

exercise grades and higher exam pass rate. Taking into account all the considerations, the adopted teaching methodology has been slightly more effective in case of Microcomputer Systems course.

Therefore, the approach seems promising, showing that consideration of current real-world problems within computer science courses increases motivation and interest among participants, leading to better overall results.

ACKNOWLEDGMENT

This work has been supported by the Ministry of Education, Science and Technological Development of the Republic of Serbia.

REFERENCES

[1] Q. Wang, M. Su., M. Zhang, R. Li, “Integrating Digital Technologies and Public Health to Fight Covid-19 Pandemic: Key Technologies, Applications, Challenges and Outlook of Digital Healthcare”, International Journal of Environmental Research and Public Health. 2021; 18(11):6053, 2021.

[2] N. Petrović, Đ. Kocić, “Smart technologies for COVID-19 indoor monitoring”, Viruses, Bacteria and Fungi in the Built Environment, ISBN: 9780323852067, Woodhead Publishing, pp. 251-272, 2021. https://doi.org/10.1016/B978-0-323-85206-7.00012-5

[3] N. Petrović, Đ. Kocić, “IoT-based System for COVID-19 Indoor Safety Monitoring”, IcETRAN 2020, pp. 603-608, 2020.

[4] N. Petrović, “Simulation Environment for Optimal Resource Planning During COVID-19 Crisis”, ICEST 2020, pp. 23-26, 2020. https://doi.org/10.1109/ICEST49890.2020.9232908

[5] N. Petrović, I. Al-Azzoni, "Model-Driven Approach to COVID-19 Vaccination Planning Leveraging Multi-Objective Optimization and Deep Learning", SSSS 2022, pp. 19-24, 2022.

[6] N. Petrović and Đ. Kocić, "IoT for COVID-19 Indoor Spread Prevention: Cough Detection, Air Quality Control and Contact Tracing," 2021 IEEE 32nd International Conference on Microelectronics (MIEL), 2021, pp. 297-300. https://doi.org/10.1109/MIEL52794.2021.9569099

[7] N. Petrović, "VHDL Logic Design Exercises Simulating COVID-19 Protection Systems", SSSS 2022, pp. 127-132.

[8] N. Petrović, “COVID-19 Safety Systems Design Exercises in Computer Science University Courses”, YuInfo 2021, pp. 1-6.

[9] N. Petrović, “Prototyping PIC16-based COVID-19 indoor safety solutions within microcomputer systems course”, IEEESTEC – 13th Student Projects Conference, pp. 185–189, 2020.

[10] N. Petrović, V. Roblek, N. Papachashvili, “Decision Support Based on Data Mining for Post COVID-19 Tourism Industry”, SAUM 2021, pp. 29-32, 2021.

[11] D. Heffelfinger, Java EE 8 Application Development, Packt Publishing, 2017.

[12] JUnit 4 [online], available on: https://junit.org/junit4/ , last accessed: 25/03/2022.

[13] MySQL [online], available on: https://www.mysql.com/ , last accessed: 25/03/2022.

[14] Docker [online], available on: https://www.docker.com/ , last accessed: 25/03/2022.

[15] RapidMiner [online], available on: https://rapidminer.com/ , last accessed: 25/03/2022.

[16] B. Brey, The Intel microprocessors: 8086/8088, 80186/80188, 80286, 80386, 80486, Pentium, Pentium Pro processor, Pentium II, Pentium III, Pentium 4, and Core2 with 64-bit extensions: architecture, programming, and interfacing, Pearson Prentice Hall, 2019.

An overview of software code review tools and the possibility of their application in teaching at the School of Electrical Engineering in Belgrade

Miloš Obradović, Marija Kostić, Balša Knežević, Dražen Drašković, *Member*, IEEE

***Abstract* - The use of version control tools together with the code review techniques is the basis of modern software development. In order to introduce future software engineers to these tools, as well as the process of software development, and to better prepare them for the industry work, the course "Principles of Software Engineering" was formed at the School of Electrical Engineering at the University of Belgrade. Within this course and the team project that students are doing, all the basic stages of the development of a software system are studied. One of the biggest challenges in organizing a practical team project is finding the right tool for code review. This tool should be suitable for educating future engineers, but also enable monitoring of students' progress and evaluation of the work done. This paper presents the basic needs that a software code review tool must meet in order to be suitable for use in education. An analysis of the functionalities of some of the existing code review tools has been given, as well as the possibility of applying these tools in education at the School of Electrical Engineering. The end of the paper presents a proposal for the best way to implement a tool for code review.**

***Index terms* - teaching methodology, program code review, software development.**

I. Introduction

Software has become an indispensable part of our daily lives, and our dependence on software is constantly increasing. An organization's success and reputation depend on its ability to produce and deliver reliable software [1]. Therefore, modern software development requires engineers to not only know how to program properly and effectively but also how to develop good engineering practices to make the codebase healthy and easy to maintain [2]. One of the techniques used in industrial and open-source projects, which aims to control the quality of code added to the codebase, is called code review [3]. The main goal of code review is to improve the readability and maintainability of the codebase. It is a process in which code is reviewed during design and development by someone other than the author. According to [2], a well-designed code review process provides several benefits:

- Allows a reviewer to check the "correctness" of the code change, i.e., is it possible for the change to introduce bugs into the codebase.
- Ensures the code change is comprehensible and understandable to other engineers.
- Enforces consistency across the codebase.
- Psychological and cultural benefits such as promotion of team ownership, validation, and recognition of one's work.
- Enables knowledge sharing.
- Provides a historical record of the code review itself.

Even though it is a widely recommended technique for improving software quality and increasing developers' productivity [4], across the industry, code review is far from the universal practice [2]. Nevertheless, together with the version control systems, the code review process forms the foundation of modern software development.

Future software engineers should be familiar with the tools and processes for version control and code review. Therefore, it is important for engineering students to review each other's source code. However, surprisingly, few engineering courses in universities and colleges include code review activities [5]. The paper [6] provides an overview of the courses that have introduced code review in their practical activities (homework, projects, etc.).

At the School of Electrical Engineering at the University of Belgrade (SEE-UB), the course "Principles of Software Engineering" (PSE) was designed to introduce students to the basic concepts of software engineering. The course covers various aspects of the software life cycle: specification design and user requirements, system design, selection of the most suitable software architecture, implementation, testing, documentation writing, and basic elements of software project management. At the core of this course is a team project in which students go through all phases of the development of a software system. Their activities range from writing basic functional specification and design of the system, to the final, tested and fully functional software product, the so-called release version. The implementation phase is based on creating a web-oriented software system on a monolithic or microservice architecture, using several basic architectural

Miloš Obradović is with the School of Electrical Engineering, University of Belgrade, 73 Bulevar kralja Aleksandra, 11020 Belgrade, Serbia (e-mail: milos.obradovic@etf.bg.ac.rs)

Marija Kostić is with the School of Electrical Engineering and the Innovation Center of the School of Electrical Engineering, University of Belgrade, 73 Bulevar kralja Aleksandra, 11020 Belgrade, Serbia (e-mail: marija.kostic@etf.bg.ac.rs), (https://orcid.org/ 0000-0003-4923-3748)

Balša Knežević is with the School of Electrical Engineering, University of Belgrade, 73 Bulevar kralja Aleksandra, 11020 Belgrade, Serbia (e-mail: balsa.knezevic@etf.bg.ac.rs)

Dražen Drašković is with the School of Electrical Engineering, University of Belgrade, 73 Bulevar kralja Aleksandra, 11020 Belgrade, Serbia (e-mail: drazen.draskovic@etf.bg.ac.rs), (https://orcid.org/ 0000-0003-2564-4526)

 ISBN 978-86-7466-930-3

and design patterns. In this school year, students can choose to develop their application using *CodeIgniter* or *Laravel* framework for *PHP*, or *Django* framework for *Python*.

Version control systems and code review process are studied as well. Within the team project, students learn to work in a team and to develop functional software, during the whole semester. Currently, team members are not involved in the code review and code testing process for other team members, so it is the desire of teachers that team members revise each other's program code. Thus, the author of the program code will always receive at least one or two reviews from other members of their team (or optionally members of another team).

In the third phase, students have to formally review the source code that some other team is working on. This phase aims to expand the knowledge and the programming techniques among students, both through what students see in other teams' solutions and through the feedback they receive from colleagues who have reviewed their solution.

For the course activities to be successful, it is necessary to find the best software tools that are available for use at the SEE-UB, which support the version control and code review. Several control version tools that are open-source are quite suitable for use in the course. However, it is much more difficult to find an appropriate system for code review that is publicly available, motivates students to work regularly on their project, and reduces the possibility of some team members avoiding doing their part of the project.

This paper is divided into five sections. The following section describes the process of code review and gives an overview of the functionalities of some popular code review tools. The third section outlines the basic requirements that a code review tool must fulfill to be used for teaching purposes at the SEE-UB. The fourth section presents an analysis of some existing code review tools, and the possibility of their application in teaching. The final section gives a brief conclusion of the results of this work and advice on how to independently implement software code review tools.

II. Analysis of Basic Features of Code Review Tools

Millions of software engineers around the world review source code daily. This process helps in finding and fixing bugs and increasing the quality of the codebase. Code reviews must be done in real-time and in multiple iterations. Thus, the use of the tools in the code review process must be simple and clear enough. Modern code review is characterized by being lightweight. It can be executed at many stages of software development, but it typically takes place before a code change is added to a version control codebase. Fig. 1. represents common steps of a code review process. First, the author creates a code change and submits it for review. Next, developers discuss the change and suggest fixes. This is an iterative process where the author has to deal with the suggested changes. Finally, when one or more reviewers approve the change, it can be added to the codebase. It is also possible to reject a code change [7].

Table I shows the basic features of the code review tools. Only tools that support Git version control system were considered in the analysis. Many of the analyzed tools also support other platforms. Only those functionalities related to the code review process were observed.

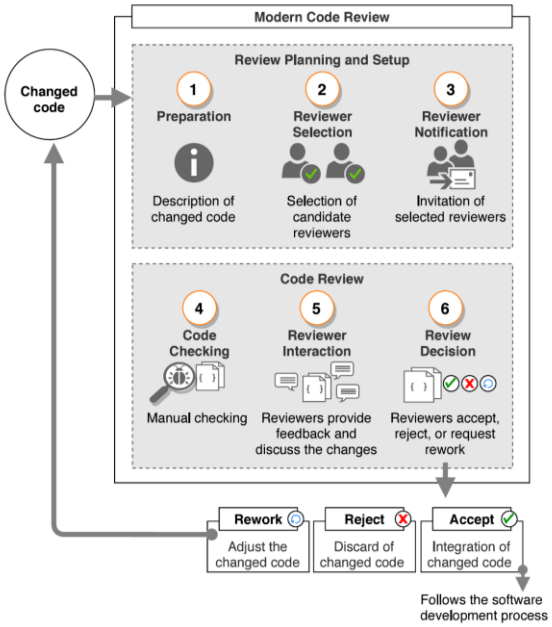


Fig. 1. Common steps of a code review process [8].

III. Necessary Code Review Tool Functionalities

This section presents in detail the basic functionalities of the code review tool that are necessary for the course. Some functionalities are important for the code review and should convey what that process looks like in practice. On the other hand, other functionalities are important for the teaching process and student collaboration on their first team project during their studies. They should motivate students to work in a team and highlight their individual programming and code review abilities. They should also provide better support for documenting a newly developed software system and commenting on different types of artifacts realized in the project phases.

As already noted, this is the first team project in their bachelor's academic studies. It is important that students cooperate well while writing a nice and readable source code, which could be upgraded later with additional modules. Currently, the team project consists of the following phases:

0. Project proposal.
1. Conceptual solution of the project with the basic functional specification.
2. Development of all use cases, one document for each functionality, and realization of the prototype in a tool for prototype development, e.g., Pencil.
3. Formal inspection of the previous phases.
4. Database modeling.
5. UML modeling of the proposed web application.
6. Implementation of a system as a web application.
7. Testing the web application.
8. Final presentation and software documentation.

 ISBN 978-86-7466-930-3

TABLE I OVERVIEW OF CODE REVIEW TOOLS AND THEIR BASIC FEATURES AND FUNCTIONALITIES.

Tool ⇨ Feature ⇩	Bitbucket Server [9]	CodeFlow [10]	Collaborator (previous version: Code Collaborator) [11]	Critique (previous version: Mondrian) [2] [12]	Crucible [13]	Gerrit (fork of Rietveld) [14] [15]	GitHub [16]	GitLab [17]	Space / Upsource [18]	Rhodecode [19]	Review Board [20]
Maintainer	*Attlassian*	*Microsoft*	*SmartBear Software*	*Google*	*Attlassian (former: Cenqua)*	*Google*	*GitHub Inc. (Microsoft)*	*GitLab Inc.*	*Jetbrains*	*RhodeCode*	*reviewboard.org*
Year of origin	2012	2009	2003	2006	2010	2009	2008	2014	2020	2010	2006
Technology stack	*Java*	N/A	N/A	N/A	*Java*	*Java* (1st ver. *Python*)	*Ruby, ECMAScript, Go, C*	*Ruby, Go, Vue.js*	*Java, Kotlin*	*Python (Pylons framework)*	*Python, Django*
License	Proprietary	Proprietary	Proprietary	Proprietary	Proprietary	Apache v2	Proprietary	MIT	Apache v2	AGPL v3	MIT
Open source	no	no	yes	no	no	yes	no	yes	no	yes	yes
Number of users	~10 million	89% of Microsoft employees	~20 000	~50000	N/A	N/A	~73 million	~30 million	N/A	N/A	N/A
Maximal repository memory capacity	4 GB	N/A	N/A	1 TB	N/A	1 TB	100 GB	10 GB	10 GB (free)	N/A	N/A
Maximal file memory capacity	1 GB	N/A	N/A	N/A	N/A	set by admin	2 GB	10 GB	10 GB	10 GB	N/A
Repository privacy	Up to 5 private-free, unlimited public	N/A	N/A	N/A	N/A	Unlimited local repo.	Unlimited public and private	Unlimited	Unlimited	N/A	Private for a fee and public
Solution type	Web-based	Standalone	Web-based	Web-based	Web-based	Web-based	Web-based	Web-based	Standalone, Web-based	Standalone, Web-based	Web-based
Version control systems support	Git, Mercurial	Git	Git, SVN, TFS, Perforce, CVS, ClearCase, RTC	Git, Piper	Git, Mercurial, CVS, Subversion, Perforce	Git	Git	Git	Git	Git, Mercurial, Subversion	Git, Mercurial, CVS, Subversion, Perforce, Bazzar, ClearCase, TFS, IBM Rational ClearCase, HCL
Review document, pictures and diagrams	N/A	N/A	No	Yes	N/A	Yes	No	No	N/A	No	No
Review at character level (prog.code)	No	Yes	No	Yes	No	Yes	No	No	No	No	No
Integration with project management tools	Jira, Trello	N/A	Jira	N/A	Jira	Jira	Jira, Trello	Jira	Trello	Jira, Trello, Redmine, Pivotal	Asana, I Done This, Trello
Integration with other tools	AWS, Crucible, Jenkins, Bamboo, MS Azure, Docker Hub, NPM, Sonar	Visual Studio	Eclipse, Visual Studio, IBM Rational Team Concert, MS Office, Adobe Reader	N/A	Bitbucket	N/A	AWS, Slack, CodeFrash, Semaphore, Asana, Azure, Google Cloud, Heroku, Travis	N/A	IDE (free), Google Calendar (Team), G Suite, Microsoft Office365, Teamcity, Jenkins	Jenkins, Travis CI, TeamCity, Confluence, Slack, HipChat, AppEnlight	Jenkins, Travis CI, Slack, Mattermostm CircleCI, Discord,

In the third phase, the formal inspection process is carried out according to the standard for formal inspection [8] and consists of six activities/steps: planning, product review, inspection meeting, realization of meeting minutes with the defect report and verification of minutes, work on corrections and final follow-up meeting to verify the corrected product. In this phase, one team of students performs a formal inspection to another team by checking all the files and documents made up to that point. The result of this phase are the reports on the formal inspection. These reports are sent to the author team for them to correct all identified shortcomings and defects, and synchronize files from the first two phases of the project. Such formal inspection could be carried out after some lather phases.

 ISBN 978-86-7466-930-3

In the sixth phase, the implementation of the software system, inspection of the source code is necessary, which has been optional so far. This paper aims to decide which software tool would be suitable for students to use in this phase. The functionalities necessary to be part of such a software tool are presented in the following subsections.

A. Availability of tools for use at the SEE-UB

The tool needs to be completely free to use or to have an academic license. Big corporations usually develop their own tools for code review, but do not make them publicly available. Other companies are actively working on the tools for software development, but their solutions are expensive, especially taking into consideration that tool will be used by several hundred students. As the code review techniques are primarily used in the industry, and less as a means of education within academic institutions [5], in the rest of this paper, only open-source tools are considered. Among the most well-known tools that offered packages with academic licenses, the following tools *GitHub*, *Atlassian Bitbucket* and *JetBrains Space* are discussed in Table 1.

B. Roles within the program code review process

The code review largely depends on the participants in the process itself. Currently, there is no universal standard in the software industry that defines exactly which people should check every change in a project [21][22]. Each company defines its own procedures for the reviewer selection process (Fig. 1, step 2). As part of the code review, there are also persons who are only in charge of checking the style of writing the program code, but not for checking the correctness of the functionality of the code [21][22].

For the course PSE there are several required functionalities related to the distribution of different roles:

- Within a single project, all students can participate in the development of the software solution, and all of them are required to review each code change, written by another team member.
- For some changes and monitoring of the program code, it should be possible to add students from other teams as code reviewers. It is necessary to enable manual addition of reviewers or addition based on the programming language/framework. As projects are developed in different programming languages (three different frameworks), it is good that the student, a member of another team, is sufficiently familiar with the syntax of that programming language, in which the authors developed their system. Comments by reviewers more familiar with the code, will be much more useful for the author [8].
- Teaching staff should be automatically added to all the teams. They should be able to just follow students' work without the obligation to review all their code changes, or with possibility to write their own reviews.

C. Possibility of anonymous code review

Research shows that significant discrimination occurs during code review when the authors of the program code or the authors of other documents are known. Discrimination can be based on gender, race, nationality, or age [22]. Additionally, in the school environment, it often happens that students with a lower average grade are afraid to criticize or point out some mistakes of students with higher grades. To motivate students to take the code review process as seriously as possible, as an important part of software development, and to reduce the effects of student discrimination and shyness, the aim is to use a tool that supports anonymous code review. Also, an anonymous review would reduce the possibility of students who know each other making personal arrangements. For example, they could decide to not find many mistakes in each other's code changes to save time they are spending on the project.

D. Ability to check the style of the program code before the code review

Static program code analysis techniques check the structure of the source code without having to execute the program itself. Their objective is to find defects early in the development process. This approach dramatically accelerates the code review process because reviewers can now focus only on the functionality and implementation of the code segment [23]. Some examples of issues that static analyzers can detect are constant expressions that overflow, uninitialized variables, tests that are never run, etc. Next to finding bugs, these tools can help verify that the code is following best practices, style guides, naming conventions, etc. in order to prevent or reduce technical debt [2]. The procedures that would be used within the project may include: writing unit tests and code coverage techniques, by each team member, for their developed parts of the program code, specific styles and precisely defined types of comments, prescribed by course teachers, etc.

As static program code analyzers reduce the cost of software development, many development environments today use them extensively. However, in the case of the course PSE, students have the freedom to choose any available development environment. A static code analyzer that supports *PHP* and *Python* programming languages is required to successfully realize the project within this course.

E. Ability to use tools to review types of files other than the program code

As already mentioned, through the course PSE, students are learning about different phases of software development. Some of the activities students face for the first time are: writing basic functional specifications, developing use case scenarios, testing web applications, and writing appropriate documentation. Next to the source code, students have to work on and produce documents of many different types. Therefore, the tool for code review should support reviewing files that are not source code. The relevant additional types of files are documents, images or specific diagrams, and other multimedia files.

 ISBN 978-86-7466-930-3

F. Ability for the teacher to access basic statistics on tool usage and change the basic code review settings

Teachers should have privileges and advanced functionalities available to them. One of these functionalities is the possibility to set up the code review process. This means that teachers should be able to form teams, control whether anonymous code reviewing is active and determine who are required reviews of some projects. Different statistics about students' work, and contributions should be also available. All this will help in further improving teaching by finding the best setup for the code review process that maximally engages students.

IV. TESTING POPULAR AND AVAILABLE CODE REVIEW TOOLS

This section shows the test results of some popular software code review tools. As explained in subsection A of the third section, for a tool to be used in teaching at the SEE-UB, it must be free for public use. The work of the following tools that meet this condition was tested in detail: *Gerrit*, *GitHub*, and *GitLab*.

Testing of three popular and publicly available tools was performed as follows. The first step is to create a project and make it available on a version control system. After that, it is necessary to do a basic review of the program code within which the tools are tested with basic files for web page structure (*html*), web page layout (*css*), program code files (*js*, *py*, and *php*) and files with commands to work with the database (*sql*). In the next step, the behavior of the code review tool is tested when files that do not contain program code were added to the project. In this step, the behavior of the tools is tested when basic document types (docx and pdf), images (*jpg* and *png*), and diagrams (*uml*) are added to the project. In the last phase of testing the software code review tool, it was checked whether the tool supports other functionalities described in the third chapter.

A. Availability, integration with Git systems, and the possibility to extend functionalities

All three tools have a basic version that is free for public use. *Gerrit* and *Gitlab* have been developed based on open-source code, which opens opportunities to independently upgrade the platform and add new functionalities per personal requirements. *GitHub*, on the other hand, is a closed-source environment, but it has a free version and offers good support for developing open-source projects. The only way to add new features to *GitHub* can be achieved through browser plugins. This is not always a good solution because it adds another item that students must install and use properly. *GitHub* also offers an academic license that brings additional functionality, but at the time of writing, the authors have not been able to go through all the necessary steps to obtain this license, so it was not possible to test its usefulness.

Code review is done through the web interface for all three tools, while support for version control is done through command line. In addition to this, the *GitHub* platform provides the ability to manage versions using the standalone application which can further facilitate the education of students in the use of software development tools. It should also be noted that the *Gerrit* platform has an interface that is not adapted for beginners and has a higher learning curve.

The *Gerrit* platform can be run on a local machine and then all data is stored on that machine. The disadvantage of this approach is that the host machine must always be available and regularly backed up data so that students do not lose their projects due to hardware failure. The advantage of this approach is the ease of adding new functionalities to the tool because the complete code is executed on a local machine.

Gerrit has a slightly different flow control compared to standard *Git* systems. Control has been simplified, which on the one hand may be good for educating new engineers, but on the other hand, it does not follow industry standards, and migrating to another code review git-based system would require adaptation.

B. Working with different file types

As expected, none of the tools had problems working with the program code. All tools provide the ability to comment on each line of program code and set the appropriate status of the comment which indicates that the code is approved or needs to be changed before getting approved.

Problems occur when the tools are used with alternative files such as documents, images, and diagrams. All three tools can display the image, but they are not able to correctly display any type of document. *Gerrit* tool provides a feature to comment on the entire file, while the other two tools do not provide this option. As a result, using *GitHub* and *GitLab*, there is no way to comment on added documents and images.

Diagrams cannot be displayed by any of these tools, but they can display the xml structure that is in the background of this file. The conclusion is that the best way to work with diagrams during software development is to attach an image with each diagram that can be commented.

C. Roles and their permissions within projects and the possibility of anonymous code review

Gerrit allows you to organize users into different groups [14]. It is possible to add users to each group individually, who will have all the authorizations assigned to that group. Using these groups, the teacher can create a project within which they will define groups and their privileges. The teacher also can create subprojects, add students to them and give them predefined group privileges.

GitLab and *GitHub* have predefined roles and access rights within the project. The roles allow the members working on the project to function well when developing software, but do not provide any additional benefits for the teacher role.

No platform has support for anonymous code review. One way to implement an anonymous code review is to make projects publicly visible. Then someone from the other team can look at the program code and submit their remarks externally to the authors of the project. Anonymous review can also be done through web plugins, but one of the problems with using anonymous review through external plugins is that all reviewers must be added to the project,

 ISBN 978-86-7466-930-3

which indicates who the potential reviewers are and reduces the effectiveness of anonymous review. Another problem is relying on the students to use external add-ons correctly, which is not easy to check, so such solutions are not the best.

All three platforms provide basic pre-processor code verification capability before review. As no style guide is currently defined in the course PSE, detailed possibilities of this functionality have not been examined in this paper and are the subject of future research.

Table II maps the functionalities required for teaching the course to the three most popular code review tools, which were publicly available.

TABLE II
ANALYSIS OF FUNCTIONALITIES OF CODE REVIEW TOOLS FOR THE NEEDS OF TEACHING THE COURSE PRINCIPLES OF SOFTWARE ENGINEERING

	Gerrit	*GitLab*	*GitHub*
Public availability	Yes	Yes	Yes
Roles in Code Review	Anonymous Change Owner, Project Owner, Registred User, Custom Gruop	Guest, Reporter, Developer, Maintainer, Owner	Read, Triage, Write, Maintain, Admin
Anonymous code review	No	No	No
Static code analysis	No	Python, PHP	Python
Review files with code	Yes	Yes	Yes
Review document, pictures and diagrams	Can comment on hole file	No	No
Automated user statistics	Available through plugins	Yes	Yes
Project configuration	Yes	Yes	Yes

V. CONCLUSION

This research provides an overview of all the commonly used code review tools in the software industry. Based on this research, the authors recommend the use of *GitLab* software, because it has user-friendly interfaces, is easy to use, has a built-in *Git* version control system, and is based on open-source code. The main disadvantage of using this platform is the limit of up to ten users per project when using the free version.

If you want to develop your code review tool and run it on a local server, then the authors recommend that you start with the *Gerrit* platform, run it on a *Linux* server and add all the features you may need locally. All analyzed tools have some shortcomings and none of them meet all the requirements for application within the course that was analyzed in this paper.

ACKNOWLEDGMENT

This paper is the result of research on the project AVANTES funded by the Science Fund of the Republic of Serbia, within the Program for the development of projects in the field of artificial intelligence. The authors are grateful for the financial support.

REFERENCES

[1] Y.-M. Zhu, “Software Reading Techniques: Twenty Techniques for More Effective Software Review and Inspection,” Solon: Apress, 2016.

[2] T. Winters, T. Manshreck and H. Wright, “Software Engineering at Google Lessons Learned from Programming Over Time,” O'Reilly Media Inc., 2020.

[3] F. A. Ackerman, L. S. buchwald and F. H. Lewski, “Software inspections: An effective verification process.,” *IEEE Software 6,,* vol. 6, no. 3, pp. 31-36, 1989.

[4] Y. K. Wong, “Modern Software Review: Techniques and Technologies,” IRM Press, 2006.

[5] V. Garousi, “Applying Peer Reviews in Software Engineering Education: An Experiment and Lessons Learned,” *IEEE Transactions on Education,* vol. 53, no. 2, pp. 182-193, 2010.

[6] T. D. Indriasari, A. Luxton-Reilly and P. Denny, “A review of peer code review in higher education,” *ACM Transactions on Computing Education (TOCE),* vol. 20, no. 3, pp. 1-25, 2020.

[7] P. C. Rigby and C. Bird, “Convergent Contemporary Software Peer Review Practices,” in *Proceedings of the 2013 9th Joint Meeting on Foundations of Software Engineering*, Saint Petersburg, 2013.

[8] N. Davila and I. Nunes, “A systematic literature review and taxonomy of modern code review,” *Journal of Systems and Software,* vol. 177, 2021.

[8] M. Greiler, “How Code Reviews work at Microsoft,” 22 January 2022. [Online]. Available: https://www.michaelagreiler.com/code-reviews-at-microsoft-how-to-code-review-at-a-large-software-company/.

[9] “BitBucket Code Review,” [Online]. Available: https://bitbucket.org/product/features/code-review. [Accessed: 03 2022].

[10] C. Staff, “CodeFlow: Improving the Code Review Process at Microsoft,” *Communications of the ACM,* vol. 62, no. 2, pp. 36-44, 2019.

[11] “Collaborator - Review Code Together,” [Online]. Available: https://smartbear.com/product/collaborator/overview/. [Accessed: 03 2022].

[12] “Google Mondrian - web-based code review and storage,” [Online]. Available: https://www.niallkennedy.com/blog/2006/11/google-mondrian.html. [Accessed: 11 2021].

[13] “Crucible,” [Online]. Available: https://www.atlassian.com/software/crucible. [Accessed: 3 2022].

[14] “Gerrit - Code Review tool,” [Online]. Available: https://www.gerritcodereview.com/. [Accessed: 12 2021].

[15] L. Milanesio, “Learning Gerrit Code Review,” Birmingham, UK: Packt Publishing Ltd., 2013.

[16] “GitHub - Code Review process,” [Online]. Available: https://github.com/features/code-review. [Accessed: 12 2021].

[17] “GitLab - Code Review Guidelines,” [Online]. Available: https://docs.gitlab.com/ee/development/code_review.html. [Accessed: 12 2021].

[18] “Space - Review Code,” [Online]. Available: https://www.jetbrains.com/help/space/review-code.html. [Accessed: 4 2022].

[19] “RhodeCode - Code Review,” [Online]. Available: https://rhodecode.com/features/productivity. [Accessed: 11 2021].

[20] “Review Board - code and document review tool,” [Online]. Available: https://www.reviewboard.org/. [Accessed: 12 2021].

[21] A. Bosu, M. Greiler and C. Bird, “Characteristics of Useful Code Reviews: An Empirical Study at Microsoft,” in *Proceedings of the International Conference on Mining Software Repositories*, Florence, Italy, 2015.

[22] E. Murphy-Hill, J. Dicker, M. Hodges, C. Egelman, C. Jaspan, L. Cheng, E. Kammer, B. Holtz, M. Jorde, A. Dolan and C. Green, “Engineering Impacts of Anonymous Author Code Review: A Field Experiment,” *IEEE Transactions on Software Engineering,* 2021.

[23] V. Balachandran, “Reducing human effort and improving quality in peer code reviews using automatic static analysis and reviewer recommendation,” in *2013 35th International Conference on Software Engineering (ICSE)*, 2013.

Automatsko snimanje amplitudno-frekventnih karakteristika primenom Arduino okruženja

Goran Dikić i Slobodan Drašković

***Apstrakt*—U radu je prikazan uređaj za automatsko snimanje amplitudno-frekventnih karakteristika elektronskih modula kao i sistema u celini. Uređaj je napravljen primenom gotovih modula za generisanje signala i merenje njihovog nivoa u radiofrekvencijskom opsegu. Proces snimanja, obrade signala kao i prikaz rezultata je automatizovan primenom Arduino NANO okruženja.**

***Ključne reči*—automatsko merenje; amplitudno-frekventne karakteristike; mikrokontroler.**

I. UVOD

U postojećoj literaturi predložena su rešenja koja omogućavaju automatsko merenje amplitudno-frekventnih karakteristika metodom koja se zasniva na primeni modernog generatora funkcija i digitalnog osciloskopa [1] ili primenom LabVIEW integrisanog razvojnog okruženja uz obradu podataka na računaru [2].

U ovom radu je prikazan uređaj za automatsko snimanje amplitudno-frekventnih karakteristika elektronskih modula ili sistema u celini koji je nastao kao rezultat potrebe da se studentima osnovnih strukovnih studija elektrotehnike i računarstva prikaže mogućnost realizacije automatizovanih mernih sistema primenom mikrokontrolera. Imajući u vidu potrebe edukativnog procesa zahtevi u pogledu programske podrške i složenosti hardvera su optimizirani tako da se kompletna analiza kao i konačna demonstracija njegovog rada mogu realizovati tokom redovnog nastavnog procesa.

Polazeći od namene jasno je da, pored mikrokontrolera, uređaj mora da sadrži generator pobudnog signala, modul za merenje nivoa signala na izlazu ispitivanog modula ili sistema kao i da ima mogućnost prikaza amplitudno-frekventne karakteristike.

Pri odabiru hardvera pošlo se od činjenice da na tržištu postoji mnoštvo elektronskih modula koji su razvijeni za rad uz podršku mikrokontrolera. S obzirom da je pretpostavljen razvoj uređaja za snimanje amplitudno-frekventnih krakteristika u radiofrekvencijskom opsegu za merenje je odabran elektronski modul koji je realizovan primenom logaritamskog pojačavača AD8307 [3]. Jedan primer njegove primene u praksi opisan je u [4].

Goran Dikić – Akademija tehničko-umetničkih strukovnih studija Beograd, Odsek Visoka škola elektrotehnike i računarstva, Vojvode Stepe 283, 11000 Beograd, Srbija, (e-mail: goran.dikic@viser.edu.rs).

Slobodan Drašković – Akademija tehničko-umetničkih strukovnih studija Beograd, Odsek Visoka škola elektrotehnike i računarstva, Vojvode Stepe 283, 11000 Beograd, Srbija, (e-mail: slobodan.draskovic@gs.viser.edu.rs).

Za generisanje pobudnog signala odabran je modul zasnovan na primeni integrisanog kola Si5351 [5]. Primer programabilnog oscilatora na bazi ovog integrisanog kola opisan je u [6].

Imajući u vidu dostupnost i karakteristike programske podrške, posebno u pogledu mogućnosti prikaza grafičkih rezultata, za realizaciju upravljanja i obradu rezultata merenja odabrano je Arduino NANO razvojno okruženje [7].

II. OPIS UREĐAJA

Logaritamski pojačavač AD8307 omogućava konverziju nivoa merenog naizmeničnog signala u odgovarajući nivo jednosmernog napona tako da se konačni rezultat može izraziti u decibelima. U konkretnom slučaju nagib merne karakteristike odgovara nivou 25 mV/dB. Pri tome je obezbeđena linearnost u opsegu ±1 dB. Dinamički opseg pojačavača je od -75 dBm do 17 dBm. Radni napon pojačavača je u opsegu od 2.7 V do 5.5 V. Izgled modula sa ugrađenim integrisanim kolom AD8307 i pripadajućim elektronskim komponentatma dat je na slici 1.

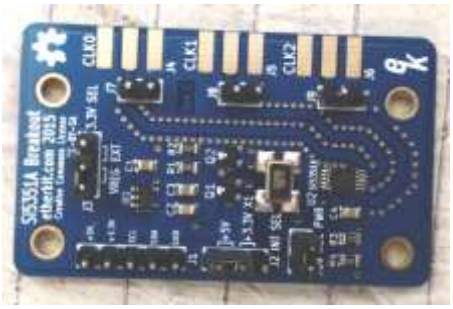
Sl. 1. Modul sa logaritamskim pojačavačem AD8307.

Modul za generisanje naizmeničnog pobudnog signala zasnovan na primeni integrisanog kola Si5351 može istovremeno da generiše tri signala na svojim izlazima CLK0, CLK1 i CLK2, ali se u konkretnom slučaju koristi samo CLK0. Generisani signal može imati frekvenciju u opsegu od 8 kHz do 160 MHz. Odabir radne frekvencije se vrši na osnovu kontrolnih signala koje šalje mikrokontroler. Pri tome se koristi I2C (*Inter-Integrated Circuit*) protokol. Integrisano kolo zahteva napajanje od 3.3 V. Modul je projektovan tako da se po potrebi može koristiti ovaj ili napon od 5 V. Izgled modula je prikazan na slici 2.

Rad celokupnog uređaja kontroliše mikrokontroler ATmega328P koji čini osnovu razvojnog sistema poznatog pod nazivom Arduino NANO [7]. Programska podrška koja je

 ISBN 978-86-7466-930-3

razvijena za rad sa ovim razvojnim sistemom pored razvoja i upisa programa u memoriju mikrokontrolera omogućava i upotrebu ekrana računara za grafički prikaz rezultata obrade tokom izvršenja konkretnog programa. Ova mogućnost je pojednostavila dizajn uređaja jer nije potrebno razvijati dodatnu programsku podršku da bi se omogućio prikaz amplitudno-frekventne karakteristike za ispitivani modul.



Sl. 2. Razvojni modul sa integrisanim kolom Si5351.

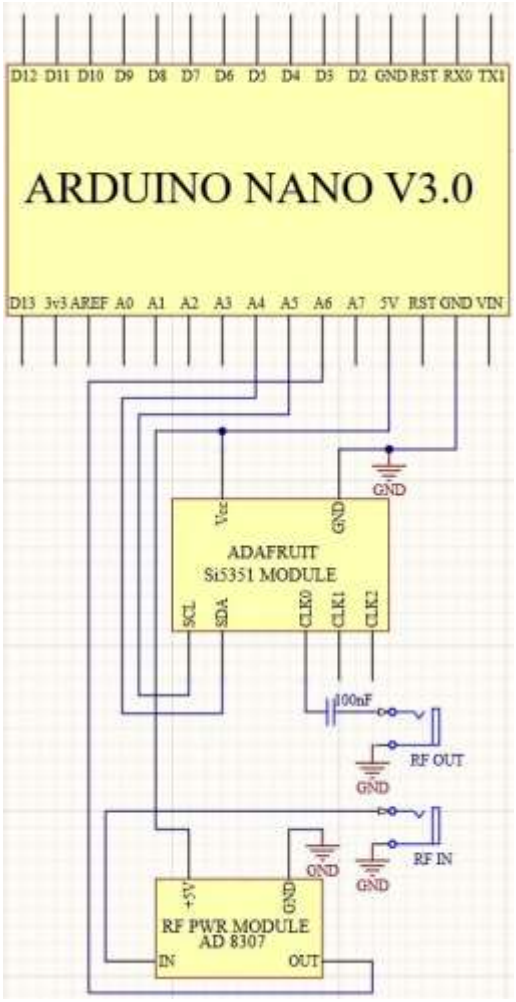
Uređaj funkcioniše tako što se u odgovarajućem programu, pre upisa u mikrokontroler, definišu početna i krajnja frekvencija na kojoj se realizuje snimanje. Veličina koraka, pri promeni frekvencije, određuje se automatski, tokom izvršenja programa, uvažavajući činjenicu da je grafička podrška dizajnirana za jednovremeni prikaz 500 tačaka na ekranu računara. Šematske veze između pojedinih modula su prikazane na slici 3.

Prenos kontrolnih signala SDA (*Serial Data and Address*) i SCL (*Serial Clock line*) ka modulu sa integrisanim kolom Si5351 ostvaruje se preko izlaza A4 i A5 Arduino NANO sistema. Na izlazu CLK0 generiše se povorka unipolarnih pravougaonih impulsa sa odnosom signal pauza u razmeri 1:1. Signal se dalje vodi preko kondenzatora od 100 nF tako da je na izlazu RF OUT potisnuta jednosmerna komponenta. Ulaz ispitivanog modula se spaja na RF OUT, a izlaz na RF IN koji zapravo predstavlja ulaz u modul sa logaritamskim pojačavačem AD8307. Izlaz ovog modula spojen je na priključak A6 Arduino NANO sistema koji predstavlja ulaz u desetobitni analogno-digitalni konvertor.

Rad uređaja, nakon upisa programa, započinje procesom definisanja neophodnih promenljivih i inicijalizacijom zahtevanih vrednosti za rad navedenih modula u skladu sa zahtevima opisanim u [3] i [5]. Realizacija i obrada svih merenja odvija se unutar programske petlje tako što se, u svakom koraku, najpre uspostavi zahtevana frekvencija signala na izlazu CLK0, a zatim sledi osam uzastopnih merenja nivoa signala na izlazu ispitivanog modula. Izmereni nivoi se obrađuju tako što se najpre saberu njihovi binarni ekvivalenti u vidu celobrojnih vrednosti. Nakon toga, dobijena vrednost se pomera unutar odgovarajućeg registra za tri mesta u desno, što odgovara deljenju sa 8. Ovim jednostavnim postupkom smanjuje se nivo šuma merenja. Dobijena vrednost se deli sa 1023 (desetobitni AD konvertor) i množi sa vrednošću referentnog napona. U ovom slučaju to je radni napon koji se dobija preko Mini-B USB (*Universal Serial Bus*) priključka Arduino NANO sistema, odnosno sa USB priključka računara. Konačna vrednost, *Vout*, koristi se za izračunavanje nivoa signala na izlazu ispitivanog modula, *PowerdB*, u decibelima:

$$PowerdB = Slope \cdot Vout - Intercept. \quad (1)$$

Pri tome *Slope* predstavlja nagib merne karakteristike, a *Intercept* vrednost presečne tačke na logaritamskoj osi [1].

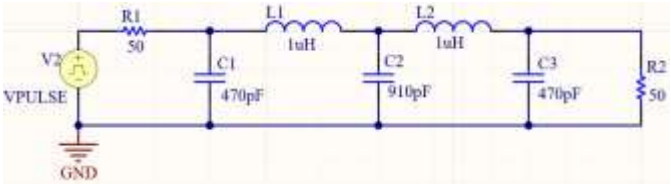


Sl. 3. Električna šema sistema za snimanje amplitudno-frekventnih karakteritika.

Rad uređaja je testiran merenjem amplitudno-frekventne karakteristike dva elektronska modula. U prvom slučaju odabran je niskopropusni LC filtar čija šema je prikazana na slici 4. Gornja granična frekvencija ovog filra je 7.2 MHz, a ulazna i izlazna impedansa su 50 Ω.

U drugom slučaju izmerena je amplutno-frekventna karakteristika kristalnog filtra 14 E 2.48 koji je proizveden u Institutu "Mihajlo Pupin", u Beogradu. Njegova centralna frekvencija je 9.0 MHz, a širina propusnog opsega, na nivou slabljenja -6 dB, iznosi ±1.2 kHz. Talasnost u propusnom opsegu je 2 dB, a slabljenje pogonskog signala nije veće od 4 dB.

 ISBN 978-86-7466-930-3

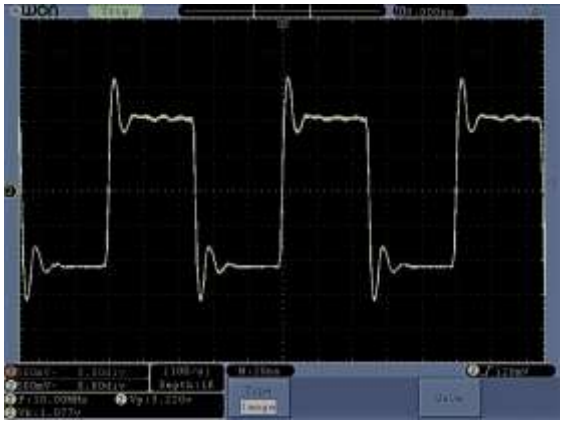


Sl. 4. Šematski prikaz niskopropusnog filtra sa graničnom frekvencijom 7,2 MHz.

Za merenje karakteristike kristalnog filtra od posebnog značaja je poznavanje njegove završne impedanse koja iznosi 390 Ω u paraleli sa kapacitivnošću od 25 pF. Važnost ovog podatka proističe iz potrebe da se obezbedi adekvatno prilagođenje impedanse kako na ulazu tako i na njegovom izlazu. U slučaju lošeg prilagođenja dobijaju se amplitudno-frekventne karakteristike koje zapravo ne odgovaraju stvarnim karakteristikama filtra (na primer velika talasnost u propusnom opsegu).

III. Rezultati eksperimenata

Prikupljanje eksperimentalnih rezultata započeto je analizom signala koji se dobija na izlazu CLK0 modula sa generatorm signala. Merenje je obavljeno na frekvenciji 10 MHz. Iz snimka koji se vidi na slici 5 uočava se postojanje preskoka kao i jednakost u pogledu trajanja poluperioda (odnos 1:1).

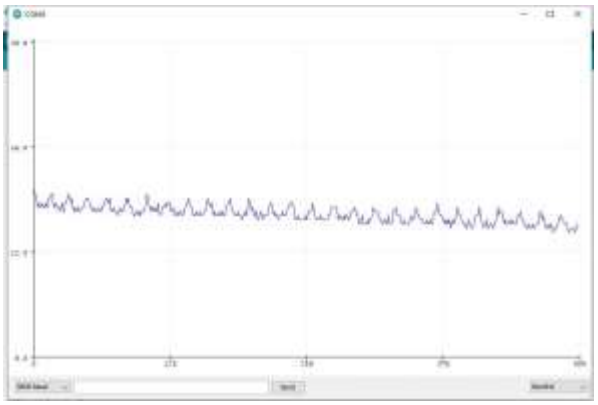


Sl. 5. Oblik signala na izlazu CLK0 (Si5351) pri frekvenciji 10 MHz.

Pravougli oblik signala nije poželjan pri merenju amplitudno-frekventnih kakrakteristika zbog činjenice da sadrži brojne harmonike, naročito kada se ima u vidu postojanje uskog i naglašenog preskoka. Sve ovo dovodi do toga da se istovremeno ostvaruje pobuda ispitivanog modula na više frekvencija pa to može uticati na kvalitet dobijenih rezultata. U svakom slučaju ovo treba imati u vidu.

U cilju sticanja uvida u nivo generisanog signala snimljena je karakteristika u opsegu 1-21 MHz, a dobijeni rezultati su prikazani na slici 6.

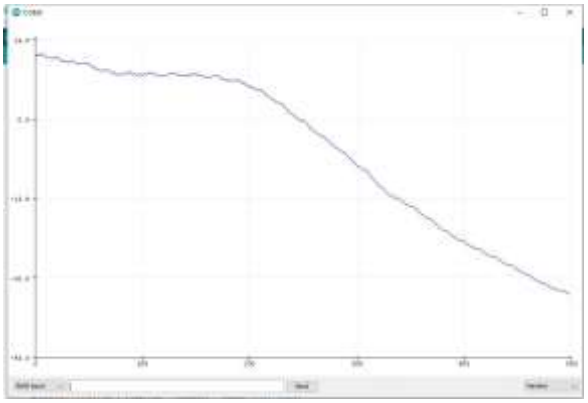
Snimanje je ostvareno direktnim povezivanjem izlaza RF OUT i ulaza RF IN. Uočeno je da nivo pobudnog signala neznatno opada sa porastom frekvencije i pokazuje izvesnu periodičnost u pogledu promena svog nivoa (pojava lokalnih ekstrema u pravilnim razmacima). Numeričke vrednosti na vertikalnoj osi dobijenog dijagrama predstavljaju nivo merenog signala u dBm. Pri tome numeričke vrednosti na horizontalnoj osi označavaju redni broj odbirka, a ne frekvenciju na kojoj je izvršeno testiranje. Ovo nije moguće menjati jer originalna programska podrška ne nudi mogućnost izbora vrednosti koje se koriste za označavanje na horizontalnoj osi. U slučaju "priliva" većeg broja podataka ceo dijagram se lagano pomera u levo tako da se na ekranu uvek može videti poslednjih 500 tačaka.



Sl. 6. Signal na izlazu Si5351 u opsegu od 1 MHz do 21 MHz.

A. *Izmerena karakteristika niskopropusnog filtra*

Očekuje se da karakteristika ispitivanog niskopropusnog LC filtra ima ravan ili blago talasasti oblik u propusnom opsegu. Dobijeni rezultati su prikazani na slici 7.



Sl. 7. Amplitudno-frekventna karakteristika niskopropusnog filra u opsegu od 1 MHz do 21 MHz.

Već je naglašeno da pobudni signal, zbog svog pravouglog oblika, sadrži mnoštvo harmonika. Svaki od njih, prolaskom kroz niskopropusni filtar učestvuje u formiranju nivoa izlaznog signala koji se meri. Porastom frekvencije pobudnog signala harmonici postaju sve udaljeniji u odnosu na njegovu osnovnu frekvenciju, odnosno sve dublje ulaze u područje gde

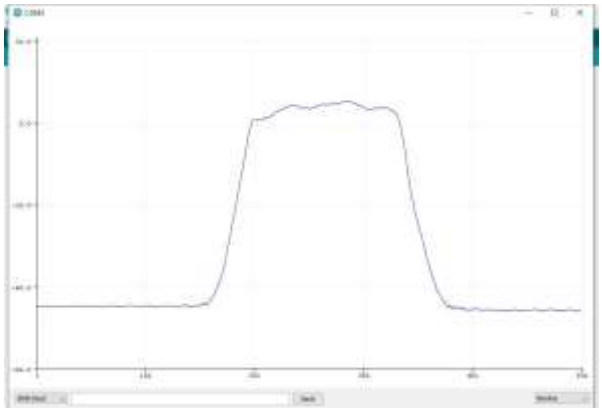
 ISBN 978-86-7466-930-3

niskopropusni filtar ostvaruje njihovo značajno slabljenje. Ovo je razlog za pojavu slabljenja (opadajući trend amplitudno-frekventne karakteristike) u početnom delu frekvencijskog opsega, bez obzira na ujednačenost nivoa pobudnog signala na ovim frekvencijama.

Nakon dovoljno visoke frekvencije pobudnog signala doprinos viših harmonika, zbog slabljenja koje unosi niskopropusni filtar, postaje zanemariv u odnosu na nivo osnovnog harmonika. Tada, karakteristika postaje približno ravna sve do prelomne frekvencije kada se, u skladu sa očekivanjem, uočava naglo slabljenje.

B. Izmerena karakteristika kristalnog filtra

S obzirom da su izlazna impedansa generatora i ulazna impedansa modula sa logaritamskim pojačavačem 50 Ω filter je spojen preko odgovarajućih transformatora sa odnosima namotaja primar sekundar u razmeri 1:8 na ulazu i 8:1 na svom izlazu. Snimanje karakteristike za kristalni filtar ostvareno je promenom frekvencije pobudnog signala u opsegu od ±5 kHz u odnosu na centralnu frekvenciju filtra (9.0 MHz). Oblik karakteristike je prikazan na slici 8.



Sl. 8. Amplitudno-frekventna karakteristika kristalnog filtra 14 E 2.48 (centralna frekvencija 9.0 MHz).

Imajući u vidu oblik karakteristike (nagli pad izvan propusnog opsega) kao i činjenicu da je snimanje vršeno unutar frekvencijskog opsega širine 10 kHz može se proceniti da širina frekvencijskog opsega takođe odgovara kataloškim podacima (±1.2 kHz u odnosu na centralnu frekvenciju). Slabljenje izvan propusnog opsega je zapravo veće u odnosu na izmereno, ali se ne može registrovati zbog ograničene dinamike samog pojačavača

IV. ZAKLJUČAK

Testiranjem su potvrđena očekivanja u pogledu problema koji mogu nastati zbog nesinusoidnog oblika pobudnog signala. Međutim, ukoliko se ima u vidu edukativna namena uređaja, predloženo rešenje nudi puno mogućnosti za povezivanje saznanja stečenih kroz izučavanje raznih predmeta i predstavlja dobru osnovu za dalji razvoj.

Za bolje rezultate trebalo bi najpre uvesti spoljni izvor referentnog napona od 2.5 V (priključak AREF na Arduino NANO modulu). Time bi se postigla bolja rezolucija, a samim tim i preciznija merenja. Bolji prikaz i očitavanje vrednosti na ekranu računara mogu se dobiti primenom MATLAB programskog okruženja (mogućnost definisanja logaritamske skale na horizontalnoj osi i ispis vrednosti koje bi označavale frekvenciju, a ne redni broj podatka). Pored toga, MATLAB omogućava dvosmernu komunikaciju tako da se zadavanje početne i krajnje frekvencije može definisati bez potrebe da se unose promene u programu koji je upisan u mikrokontroler. Naravno ovo izlazi iz okvira optimizacije koja je usvojena na početku razvoja uređaja, a odnosi se na maksimalno iskorišćenje originalnog ARDUINO programskog okruženja.

Konačno, moguće je ostvariti promene u pogledu proširenja hardvera dodavanjem LCD ekrana i obrtnog enkodera sa tasterom. Na taj način bi se pored zadavanja početne i krajnje frekvencije, uz uvođenje ogovarjućih poruka na LCD ekranu mogle uvesti i dodatne mogućnosti kao, na primer, zahtev da se ponovi snimanje ili da se uspostavi režim rada koji omogućava kalibraciju kompletnog uređaja prilikom prvog puštanja u rad.

LITERATURA

[1] L. Satish, Santosh C. Vora, "Amplitude Frequency Response Measurement: A Simple Technique", *IEEE Transactions on Education*, vol. 53, no. 3, pp. 365-371, Aug. 2010.

[2] Tian Tian, Wu Jian, Nie Li, "*Applied Mechanics and Materials*", *AMM*, vol. 475-476, pp.16-22, 2010.

[3] Low Cost, DC to 500 MHz, 92 dB Logarithmic Amplifier, https://www.analog.com/media/en/technical-documentation/data-sheets/ad8307.pdf, pristupljeno 17. 01. 2022.

[4] J.C. Cowles, "The Evolution of Integrated RF Power Measurement and Control", Proc. IEEE MELECON 2004, Dubrovnik, Croatia, pp. 131-134, May 12-15, 2004.

[5] Etherkit Si535, https://www.arduino.cc/reference/en/libraries/etherkit-si5351/, pristupljeno 27. 01. 2022.

[6] Janko Koležnik, Boštjan Vlaovič, "Programabilni kristalni oscilator visoke razločljivosti", *Elektrotehniški Vestnik*, vol. 84, no. 3, pp. 93-98, 2010.

[7] Arduino Nano, https://docs.arduino.cc/hardware/nano, pristupljeno 15. 12. 2022.

ABSTRACT

In this paper the device for automatic recording of amplitude-frequency characteristics of electronic modules as well as whole system is presented. The device is designed applying readymade modules for signals generating and level measurement of them in radiofrequency band. Recording process, signal processing as well as presentation of results is automatized using Arduino NANO environment.

Automatic recording of amplitude-frequency characteristics using Arduino environment

Goran Dikić and Slobodan Drašković

 ISBN 978-86-7466-930-3

Zaštita prenosa paketskog telefonskog saobraćaja upotrebom tehnologije virtuelnih privatnih mreža

Mićo Živanović, Jovan Bajčetić, Ivan Tot

***Apstrakt*—Istraživanje predstavljeno u ovom radu prikazuje jednu realizaciju zaštite paketskog telefonskog saobraćaja primenom tehnologije virtuelnih privatnih mreža kroz konfiguraciju servera za prenos paketskog telefonskog saobraćaja i zaštićeni prenos uz primenu tehnologije virtuelnih privatnih mreža u tunel modu, primenom odgovarajućeg protokola za zaštitu tajnosti, autentifikaciju, zaštitu integriteta i razmenu kriptografskih ključeva. Izvršeno je snimanje i analiza saobraćaja primenom softvera Wireshark u zaštićenom i nezaštićenom prenosu. Prikazani rezultati omogućavaju lakše razumevanje kompleksnog procesa uspostave tunela upotrebom simulacionog softvera u edukaciji.**

***Ključne reči*—paketski telefonski saobraćaj; virtuelne privatne mreže; zaštićena komunikacija; kriptografski ključ.**

I. Uvod

Stalan razvoj Interneta ima za posledicu da je Internet postao univerzalno sredstvo za komunikaciju. U toku razvoja, postavio se zahtev za bezbednošću prenošenih informacija koji se ogledao u obezbeđenju bezbednosnih servisa: autentifikacije, poverljivosti, neporecivosti i integriteta podataka [1]. Razvijeni su sistemi zaštite u tri ravni: upravljačkoj, kontrolnoj i ravni podataka. Za potrebe ovog rada biće razmotreni mehanizmi zaštite u ravni podataka koji se odnose na informacioni saobraćaj. Ravan podataka se štiti pomoću implementiranja pravila (bezbednosnih polisa) po kojima se informacioni sadržaj prenosi upotrebom mrežnih uređaja.

Jedna od tehnologija koja omogućava zaštitu prenosa podataka u ravni podataka je tehnologija virtuelnih privatnih mreža (VPN – Virtual Private Networks). Navedena tehnologija pruža sledeće mogućnosti umrežavanja:
– Intranet, umrežavanje geografski dislociranih objekata;
– Udaljeni pristup mobilnih korisnika (rad od kuće);
– Ekstranet, ograničeni pristup nekoj mreži iz drugih mreža (pristup poslovnih partnera korporativnom WAN-u) [2].

Za realizaciju virtuelne privatne mreže mogu se koristiti periferni korisnički uređaji (host, ruter ili svič), na lokaciji korisnika (CE – Customer Edge) i periferni mrežni uređaji provajdera (PE – Provider Edge).

Virtuelnu privatnu mrežu čini više udaljenih mreža koje su povezane preko Interneta. Zbog korišćenja zajedničkih resursa na Internetu, komunikacija među korisnicima virtuelne privatne mreže se mora zaštititi. Zaštita virtuelne privatne mreže se ostvaruje pomoću barijera koje implementiraju IPSec protokol u tunel modu [3].

VPN tunel je veza između dva PE rutera ili dva CE uređaja koji predstavljaju krajnje tačke tunela [2].

Prema IETF, IP VPN se mogu klasifikovati u zavisnosti od odgovornosti u pogledu upravljanja na:
– VPN kojima upravlja korisnik (Customer Provisioned VPN, CP VPN);
– VPN kojima upravlja provajder servisa (Provider Provisioned VPN, PP VPN) [2].

Prema lokaciji VPN opreme, PP VNP se mogu podeliti na:
– CE – bazirane, kod kojih su krajnje tačke VPN locirane kod korisnika;
– PE – bazirane, kod kojih su krajnje tačke VPN tunela locirane kod provajdera, na PE ruteru.

U zavisnosti od ponuđenog servisa, PE – bazirane VPN se dele na:
– PE – bazirane L2 VPN (koje pružaju servise OSI sloja 2);
– PE – bazirane L3 VPN (koje pružaju servise OSI sloja 3);
– CE – bazirane IP VPN pružaju samo servise OSI sloja 3.

Istovremeno sa razvojem bezbednosnih servisa razvijaju se arhitekture za pružanje različitih komunikacionih servisa koji koriste Internet protokol (telefonija, video, podaci, multimedijalni servisi). Prenos telefonije preko Interneta razvijao se postupno, pre svega zbog prethodno razvijenih sistema klasičnih javnih telefonskih mreža (PSTN) i digitalnih mreža sa integrisanim servisima (ISDN).

U cilju razvoja telefonije zasnovane na komutaciji paketa (VoIP telefonija), razvijene su grupe protokola za prenos VoIP telefonije i povezivanje VoIP telefonije sa telefonijom koja se prenosi u drugim sistemima prenosa (H.323 i SIP protokol) [4].

Prikaz istraživanja u ovom radu će se sastojati iz opisa načina realizacije zaštite informacije korišćenjem VPN tehnologije, a potom u jednoj realizaciji zaštite prenosa paketskog telefonskog saobraćaja upotrebom tehnologije

Mićo Živanović – Ministarstvo odbrane, Sektor za ljudske resurse, Nemanjina 15, 11000 Beograd, Srbija (e-mail: comiveza@yahoo.com).
Jovan Bajčetić – Vojna Akademija, Univerzitet odbrane u Beogradu, Veljka Lukića Kurjaka 33, 11042 Beograd, Srbija (e-mail: bajce05@gmail.com).
Ivan Tot – Vojna Akademija, Univerzitet odbrane u Beogradu, Veljka Lukića Kurjaka 33, 11042 Beograd, Srbija (e-mail: totivan@gmail.com).

 ISBN 978-86-7466-930-3

virtuelnih privatnih mreža, upotrebom IPSec protokola u tunel modu ka udaljenom korisniku, uz prikaz analize mrežnog saobraćaja korišćenjem programskog alata Wireshark.

II. ZAŠTITA PODATAKA PRIMENOM TEHNOLOGIJE VIRTUELNIH PRIVATNIH MREŽA

Za prenos informacionog sadržaja preko Interneta neophodno je obezbediti zaštitu u prenosu. Čest je slučaj da kompanije koriste Internet kao okosnicu za povezivanje svojih filijala ili klijenata kako bi ostvarili prenos podataka za svoje potrebe. Iz navedenog razloga nameće se potreba za zaštitu prenošenog saobraćaja. U tu svrhu koriste se različite tehnologije, od kojih je jedna - tehnologija virtuelnih privatnih mreža. Za realizaciju virtuelnih privatnih mreža na raspolaganju je više tehnologija, zavisno od toga da li se VPN realizuje kao "oblast – oblast" (site–to–site) ili kao "udaljeni pristup" (remote access). U oba navedena slučaja najčešće se koristi IPSec (IP security) protokol.

IPSec protokol štiti pakete između dva uređaja u mreži [3].

Uređaji kojima se realizuje IPSec su: server, ruteri, korisnički računari ili specijalizovani hardver. IPsec pruža dve vrste zaštite: autentifikaciju i poverljivost.

Mehanizam autentifikacije osigurava da je primljeni paket zaista poslao onaj ko je u zaglavlju paketa naveden kao izvor i da se paket nije promenio tokom prenosa, dok mehanizam poverljivosti omogućava entitetima u komunikaciji da šifruju poruke kako bi sprečili nepozvana lica da dođu do sadržaja poruka [1]. Za šifrovanje podataka se koriste simetrični algoritmi (DES, 3DES, AES), što zahteva pouzdanu razmenu ključeva strana u komunikaciji i za tu svrhu se koriste protokoli za autentifikaciju (neki od protokola iz IETF (IKE - Internet Key Exchange) standarda) [3].

Razvoj bezbednosti u arhitekturi Interneta se odvijao postepeno u čemu je značajno mesto imala Radna grupa za inženjering Interneta (IETF – Internet Engineering Task Force). Sâmo uvođenje standarda je išlo postepeno. Prvi u nizu standard IETF koji se odnosio na bezbednost u arhitekturi Interneta bio je standard RFC 1636 (Request for Comments). Standard se odnosio na osnove bezbednosti Interneta (upotreba firewall – a, servis autentifikacije, privatnost i dr.) [5].

Da bi se adekvatno razumeo način formiranja VPN sesije in a pravi način predstavio prilikom edukacije, biće razmotreno nekoliko najvažnijih dokumenata IETF kojima su definisani režimi rada VPN, mehanizam autentifikacije, razmena kriptografskih ključeva i zaštita poverljivosti.

IPSec koristi dva protokola za bezbednost: AH (Authentication Header) i ESP (Encapsulating Security Payload). Zaglavlje autentifikacije (AH) je definisano specifikacijom RFC 4302 (IP Authentication Header), dok je ESP enkapsulirajuće bezbedno pakovanje (Encapsulating Security Payload) definisan specifikacijom RFC 4303. AH i ESP podržavaju dva režima rada: transportni režim i tunelovanje.

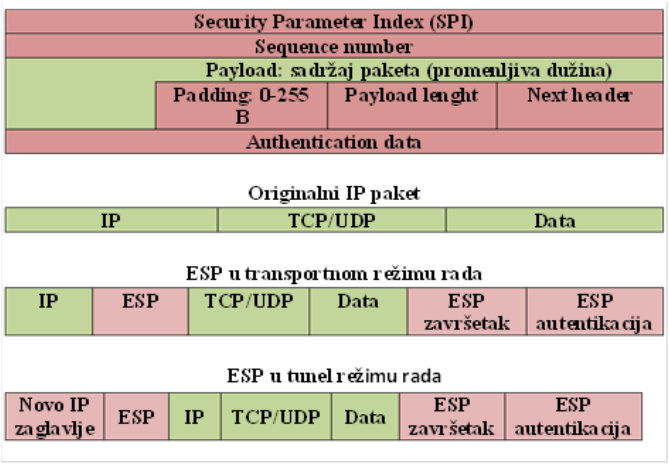
U transportnom režimu AH autentifikuje IP koristan sadržaj i odabrane delove IP zaglavlja, dok ESP šifruje i opciono autentifikuje IP koristan sadržaj. Tunelovanje vrši zaštitu celog IP paketa. Navedeno se postiže nakon dodavanja AH i ESP polja i tretiranja celog paketa kao korisnog sadržaja novog spoljnog IP paketa sa novim IP zaglavljem.

U režimu tunelovanja ESP šifruje i opciono autentifikuje ceo unutrašnji IP paket, uključujući unutrašnje IP zaglavlje, dok AH u režimu tunelovanja autentifikuje ceo unutrašnji IP paket i odabrane delove spoljnog IP zaglavlja [1].

Redosled postupaka sa paketima za rad ESP u transportnom i tunel režimu, je sledeći:

– U transport režimu blok podataka koji se sastoji od segmenta transportnog sloja sa dodatim ESP završnim blokom se šifruje, sa dodatim zaglavljem za autentifikaciju (opciono);

– U režimu tunelovanja, ESP se koristi za šifrovanje celog IP paketa, ESP zaglavlje ide ispred paketa i šifruje se paket zajedno sa ESP završnim blokom.



Sl. 1 Opseg ESP šifrovanja u transportnom i tunel modu [6]

Sl. 1 prikazuje format jednog ESP paketa. Indeks bezbednosnih parametara (SPI) definiše jednu bezbednosnu asocijaciju kojom se određuje algoritam šifrovanja i autentifikacije, ključevi, inicijalizacione vrednosti, životni vek ključeva i vezani parametri koji se koriste uz ESP. Broj sekvence je vrednost brojača paketa kojom se sprečava ponavljanje paketa. Sadržaj paketa je promenljive dužine i predstavlja segment transportnog sloja (transport režim) ili IP paket (tunel režim). U tunel modu, celom paketu se dodaje novo IP zaglavlje koje ima dovoljno informacija za rutiranje, ali ne i za analizu saobraćaja [6].

Važan deo IPSec koji se odnosi na upravljanje ključevima obuhvata određivanje i distribuciju ključeva. Dokumentom RFC 4301 definisane su dve vrste upravljanja ključevima:

– Ručno (administrator definiše sistem sopstvenim ključevima i ključevima drugih sistema sa kojima komunicira);

– Automatizovano (omogućava generisanje ključeva za bezbednosnu asocijaciju na zahtev koji je pogodan za velike sisteme sa rastućom konfiguracijom) [7].

Protokol koji se koristi za automatizovano upravljanje ključevima za IPSec je ISAKMP (Internet Security

 ISBN 978-86-7466-930-3

Association and key Management Protocol) i definisan je dokumentom IETF RFC 2408. ISAKMP definiše procedure za kreiranje i upravljanje bezbednosnim asocijacijama, tehnike generisanja ključeva, ublažavanje pretnji (npr. od DDoS napada) [8].

ISAKMP ne nalaže konkretan algoritam za razmenu ključeva, već se sastoji od jednog skupa tipova poruka koje omogućavaju upotrebu raznovrsnih algoritama za razmenu ključeva.

Karakteristike IKE određivanja ključeva su:
- Osujećenje DDoS napada;
- Omogućava razmenu ključeva za pregovaranje oko grupe ključeva;
- Obezbeđuje od napada ponavljanjem korišćenjem jednokratnih brojeva;
- Omogućava razmenu javnih ključeva;
- Onemogućava napad tipa "čovek u sredini".

IKE potprotokol obezbeđuje dogovaranje protokola, algoritama i ključeva između učesnika u komunikaciji, proverava autentičnost učesnika koji učestvuju u postupku dogovaranja, omogućava razmenu podataka na osnovu kojih će se generisati ključevi i upravljati razmenom ključeva. IKE potprotokol obavlja se u dve faze [9].

U prvoj fazi dva učesnika uspostavljaju bezbedni komunikacioni kanal kojim će se obaviti dogovaranje bezbednosnih parametara i razmena ključeva. Dogovaranje parametara i razmena ključeva, odnosno uspostava bezbednosne asocijacije obavlja se u drugoj fazi. Za sprovođenje postupka koriste se tri načina razmene informacija, dva za prvu fazu i jedan za drugu fazu IKE potprotokola:
- Osnovni način;
- Agresivni način;
- Brzi način.

Osnovni način razmene informacija (engl. Main mode) koristi se u prvoj fazi IKE potprotokola i služi da bi se uspostavio bezbednosni komunikacioni kanal kojim će se obaviti razmena podataka potrebnih za kasniju komunikaciju AH ili ESP potprotokolima.

Agresivni način razmene, slično kao i osnovni, koristi se u prvoj fazi IKE potprotokola i služi za uspostavljanje sigurnog komunikacionog kanala za dogovor učesnika i ne obavlja se kroz bezbedni kanal. Agresivni način koristi samo tri poruke u razmeni i nešto je jednostavniji i brži od osnovnog načina, ali se dokazivanje identiteta ne vrši kroz bezbedan kanal.

Nakon uspostave bezbednog kanala primenom osnovnog ili agresivnog načina razmene, započinje druga faza IKE potprotokola. Druga faza koristi se brzim načinom razmene ključeva koja služi za dogovaranje bezbednosnih parametara komunikacije AH ili ESP potprotokolom i za razmenu tajnih simetričnih ključeva.

III. JEDNA REALIZACIJA ZAŠTITE PAKETSKOG TELEFONSKOG SAOBRAĆAJA UPOTREBOM TEHNOLOGIJE VIRTUELNIH PRIVATNIH MREŽA

U uvodu rada predstavljena je podela VPN prema tome ko je odgovoran za uspostavu zaštićene komunikacije (provajder ili korisnik), kao i koja vrsta servisa se ostvaruje (sloj 2 ili 3 OSI referentnog modela). Predloženi model koje će u nastavku biti prikazan omogućava realizaciju jedne VPN koja bi predstavljala primer uspostave zaštite VoIP putem VPN za koju je „odgovoran" provajder, na OSI sloju 3 i da se primenom programskog alata „Wireshark" snimi i analizira ostvareni saobraćaj. Za navedene potrebe je uspostavljena mrežna topologija prikazana na Sl. 2.

Ruteri predstavljaju periferne rutere provajdera na kojima se vrši konfigurisanje VPN konekcije, po modelu "oblast – oblast". Na ruteru 1 su konfigurisani i uspostavljeni VPN, VoIP i DHCP server.

VPN server je konfigurisan sledećim parametrima:
- ISAKMP razmena kripto ključeva (policy 10);
- Kripto algoritam 3DES;
- Algoritam za autentifikaciju MD5 [10];
- Rad u tunel modu.

VoIP server je određen sledećim parametrima:
- Konekcija SIP protokolom;
- Kodek g711 ulaw.

Za razumevanje rada VoIP, ukratko će biti objašnjen prenos signalizacije i kontrola saobraćaja u VoIP prenosnim sistemima, u kojima se najčešće koriste H.323 i SIP protokoli.

H.323 preporuka je deo familije ITU-T preporuka sa zajedničkom oznakom H.32x koje se odnose na multimedijalne komunikacije preko različitih mreža. H.323 definiše protokole zadužene za usluge multimedijalnih komunikacija preko mreža zasnovanih na komutaciji paketa. H.323 se najčešće koristi kao signalizacioni i kontrolni protokol u VoIP i za video konferencije, a bio je prvi standard koji je koristio RTP protokol (Real-time Transfer Protocol) za konkretni prenos audio i video signala preko mreže.

H.323 je standard koji omogućava multimedijalnu komunikaciju preko različitih mreža (usko pojasne ISDN, širokopojasne B-ISDN, lokalne računarske mreže, mreže na bazi komutacije kola). Cilj je postizanje interoperabilnosti sa različitim mrežama za prenos multimedijalnih informacija, kroz upotrebu zajedničkih preporuka, procedura i poruka, kao i uvođenjem komponente mrežnog prolaza. H.323 standard predstavlja skup protokola namenjenih za obavljanje različitih funkcija u okviru H.323 sistema i to: audio kodere i dekodere, video kodere i dekodere, signaliziranje poziva, kontrola poziva, protokol prenosa u realnom vremenu (RTP), protokol kontrole prenosa u realnom vremenu (RTCP), registraciju, pristup i status i ostale protokole za prenos podataka u realnom vremenu [4].

 ISBN 978-86-7466-930-3

SIP je protokol za uspostavljanje, modifikaciju i raskidanje multimedijalnih sesija u paketskim mrežama. SIP u kombinaciji sa drugim protokolima se koristi za opis karakteristika sesije potencijalnim učesnicima [11]. SIP je delo IETF (Internet Engineering Task Force) i razvijen je kao mehanizam za uspostavljanje raznovrsnih sesija, a može se koristiti za unicast i multicast komunikaciju. SIP je peer-to-peer protokol, što znači da nije centralizovan, već je servisna inteligencija izmeštena prema krajevima mreže ka krajnjim korisnicima, kao kod računarskih mreža. U okviru SIP poruka se najčešće prenosi SDP (Session Description Protocol), mada standard ostavlja otvorenim i druge mogućnosti [12].

H.323 i SIP su dva konkurentska protokola za multimedijalne komunikacije na paketskim mrežama.

SIP se odlikuje sledećim prednostima:

– Fleksibilnost (omogućava korišćenje sa različitim transportnim i drugim protokolima);
– Arhitektura i osobine mu se prirodno uklapaju Internet okruženje, dok H.323 ima neke osobine protokola fiksne telefonije;
– Poseduje mnoga proširenja potrebna za različite sisteme ličnih komunikacija (prisutnost, instant poruke, posredno upravljanje pozivom) [13].

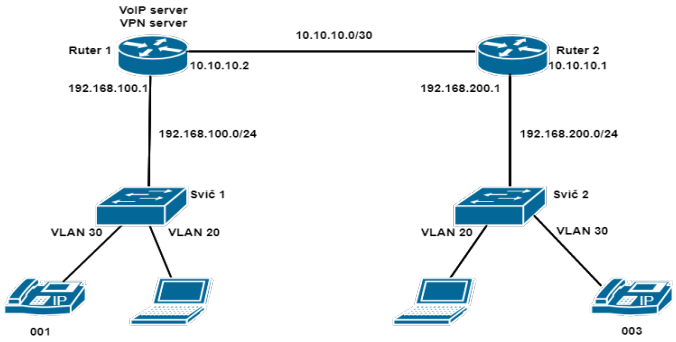
DHCP server je uspostavljen za opseg adresa koji obezbeđuje formiranje logički odvojenih mreža sa opsegom adresa 20.20.20.0 i 30.30.30.0 u različitim virtuelnim lokalnim mrežama. Prema mrežnoj topologiji formirane su dve LAN mreže u okviru kojih je izvršeno razdvajanje saobraćaja (telefonskog i podaci), uspostavom dve VLAN na OSI sloju L2, konfigurisanjem svičeva 1 i 2 (VLAN 20 i 30), dok je po jedan interfejs na svičevima konfigurisan kao trank interfejs [14]. Nakon provere konektivnosti, uspostavljen je paketski telefonski saobraćaj bez zaštite pomoću virtuelne privatne mreže i realizovano snimanje saobraćaja, upotrebom programskog alata “Wireshark”. Rezultat i procesi analize nezaštićenog saobraćaja prikazani su u Tabeli 1. Uspostavljena je VPN sesija između dva rutera i realizovano snimanje saobraćaja u zaštićenom modu. Procesi u toku uspostave zaštićenog paketskog telefonskog saobraćaja prikazani su u Tabeli 2.

Za realizaciju mrežne topologije na slici 2 korišćena je sledeća mrežna oprema:

– CISCO 2900 ruter…….2 kom;
– CATALIST 3650 svič ……..2 kom;
– Računar sa ETH mrežnim interfejsom…. 2 kom;
– IP telefoni……..2 kom.

Procesi prikazani u Tabeli 1, a koji se odnose na uspostavu i održavanje nezaštićenog telefonskog saobraćaja prikazuju proces uspostavljanja prisutnosti uređaja u mreži i utvrđivanja mrežnih usluga (SSDP), uspostavu logičke topologije mreže i saobraćaja protokola za sprečavanje petlji (STP). Pozivanjem jednog korisnika od strane drugog korisnika ustanovljava se IP adresa pozvanog korisnika kroz broadcast upit od strane pozivajućeg korisnika (ARP proces). Kroz DNS proces se povezuju IP adresa i ime domena pozvanog korisnika. Istovremeno se šalje poruka radi utvrđivanja dostupnosti korisnika (ICMP poruka). U toku uspostave VoIP komunikacije šalje se “hello poruka” u OSPF procesu, radi konstruisanja putanje između dva rutera. Kroz SIP signalizaciju vrši se pozivanje jednog od strane drugog korisnika, nakon čega se ostvaruje TCP sesija kroz proces “trostrukog rukovanja”. U sklopu SIP procesa razlikuju se faze (traying, ringing i OK), u kojima se mogu uočiti status procesa pozivanja, koji se na kraju završava uspešnom uspostavom komunikacije. Ceo proces je praćen slanjem kontrolnih TCP poruka (ACK), kojima se određuje broj bajtova koji se može poslati pre dobijanja sledeće dozvole za slanje, kao i slanjem poruka kojima se vrši sinhronizacija uspostave TCP sesije (SYN). Poseban segment kontrole komunikacije predstavlja kontrola konektivnosti na L2 nivou (LOOP). Dalji proces komuniciranja je praćen razmenom paketa u realnom vremenu (RTP) koji nose govorni informacioni sadržaj.

Procesi prikazani u Tabeli 2, a koji se odnose na uspostavu i održavanje zaštićenog telefonskog saobraćaja, pored na početku navedenih procesa u nezaštićenom prenosu, sadrži proces razmene kripto ključeva u procesu uspostave IPSec tunela (ISAKMP). Specifično za proces prenosa u zaštićenom modu je uspostavljen trank između dva sviča na kojima su konfigurisane dve VLAN (DTP).



Sl. 2 Mrežna topologija za potrebe jedne realizacije zaštite paketskog telefonskog saobraćaja upotrebom tehnologije virtuelnih privatnih mreža

Prikazana realizacija mrežne topologije, snimanje i analiza mrežnog saobraćaja omogućava detaljno razumevanje uspostave VPN tunela, uz upotrebu simulacionog softvera, kao i praćenje procesa razmene informacionog sadržaja u realnom vremenu. Tokom procesa prenosa informacionog sadržaja u realnom vremenu, periodično se uočavaju procesi uspostavljanja prisutnosti uređaja u mreži (SSDP), obaveštavanja o nedostupnosti uređaja u slučaju prekida konektivnosti (ICMP poruke) i razmena “hello” poruka u OSPF procesu. Navedena realizacija stoga omogućava potpun uvid u sve procese u toku prenosa paketskog telefonskog saobraćaja, što može poslužiti u edukaciji i o procesima prenosa multimedijalnih informacionih sadržaja.

 ISBN 978-86-7466-930-3

TABELA I
POZIV UČESNIKA 001 KA UČESNIKU 003 BEZ VPN ZAŠTITE

R.br.	Vreme [s]	Izvorišna adresa	Odredišna adresa	Protokol	Veličina [B]
1.	0,00	Cisco_e1:aa:81	Spanning_tree	STP	60
2.	0,17	Fe80:e010:c683:5106:f3e8	Ff02::c	SSDP	208
3.	1,99	Cisco_e1:aa:81	Spanning_tree	STP	60
4.	2,73	ASUSTEkC_9d:86:8a	Cisco_00:e3:e1	ARP	42
5.	2,74	Cisco_00:e3:e1	ASUSTEkC_9d:86:8a	ARP	60
6.	3,433	40.40.40.3	192.168.100.1	DNS	75
7.	3,434	40.40.40.1	40.40.40.3	ICMP	70
8.	4,00	Cisco_e1:aa:81	Spanning_tree	STP	60
9.	4,18	Fe80:e010:c683:5106:f3e8	Ff02::c	SSDP	208
10.	5,32	40.40.40.1	224.0.0.5	OSPF	90
11.	5,99	Cisco_e1:aa:81	Spanning_tree	STP	60
12.	6,56	40.40.40.3	192.168.100.1	DNS	75
13.	6,561	40.40.40.1	40.40.40.3	ICMP	70
14.	6,83	40.40.40.3	30.30.30.1	SIP/SDP	922
15.	7,16	40.40.40.3	30.30.30.1	TCP	590
16.	7,16	30.30.30.1	40.40.40.3	TCP	60
17.	7,16	40.40.40.3	30.30.30.1	TCP	386
18.	7,16	30.30.30.1	40.40.40.3	SIP	418
19.	7,18	Fe80:e010:c683:5106:f3e8	Ff02::c	SSDP	208
20.	7,36	40.40.40.3	30.30.30.1	TCP	54
21.	7,99	Cisco_e1:aa:81	Spanning_tree	STP	60
22.	8,30	40.40.40.3	30.30.30.1	TCP	58
23.	8,32	Cisco_e1:aa:81	Cisco_e1:aa:81	LOOP	60
24.	8,37	30.30.30.1	40.40.40.3	TCP	590
25.	8,37	30.30.30.1	40.40.40.3	SIP	124
26.	8,37	40.40.40.3	30.30.30.1	TCP	54
27.	9,99	Cisco_e1:aa:81	Spanning_tree	STP	60
28.	10,18	Fe80:e010:c683:5106:f3e8	Ff02::c	SSDP	208
29.	11,97	10.10.10.2	40.40.40.3	RTP	214
30.	11,97	30.30.30.1	40.40.40.3	TCP	590
31.	11,97	30.30.30.1	40.40.40.3	SIP/SDP	424
32.	11,97	40.40.40.3	30.30.30.1	TCP	54
33.	11,98	40.40.40.3	10.10.10.2	RTP	55
34.	11,98	40.40.40.3	10.10.10.2	TCP	66
35.	11,98	10.10.10.2	40.40.40.3	TCP	60
36.	11,98	40.40.40.3	10.10.10.2	TCP	54
37.	11,98	40.40.40.3	10.10.10.2	SIP	459
38.	11,98	10.10.10.2	40.40.40.3	TCP	60
39.	11,99	10.10.10.2	40.40.40.3	RTP	214
40.	11,99	Cisco_e1:aa:81	Spanning_tree	STP	60
41.	12,01	10.10.10.2	40.40.40.3	RTP	214
42.	12,03	10.10.10.2	40.40.40.3	RTP	214
43.	12,25	40.40.40.3	10.10.10.2	RTP	214

TABELA II
POZIV UČESNIKA 001 KA UČESNIKU 003 SA VPN ZAŠTITOM

R.br.	Vreme [s]	Izvorišna adresa	Odredišna adresa	Protokol	Veličina [B]
1.	0,00	Fe80:e010:c683:5106:f3e8	Ff02::c	SSDP	208
2.	0,03	Cisco_e1:aa:81	Spanning_tree	STP	60
3.	0,73	40.40.40.3	10.10.10.2	ISAKMP	126
4.	0,73	10.10.10.2	40.40.40.3	ISAKMP	126
5.	2,03	Cisco_e1:aa:81	Spanning_tree	STP	60
6.	4,00	Fe80:e010:c683:5106:f3e8	Ff02::c	SSDP	208
7.	4,03	Cisco_e1:aa:81	Spanning_tree	STP	60
8.	6,03	Cisco_e1:aa:81	Spanning_tree	STP	60
9.	6,88	Fe80:e010:c683:5106:f3e8	Ff02::1:2	DHCPv6	149
10.	7,00	Fe80:e010:c683:5106:f3e8	Ff02::c	SSDP	208
11.	8,03	Cisco_e1:aa:81	Spanning_tree	STP	60
12.	8,67	Cisco_e1:aa:81	Cisco_e1:aa:81	LOOP	60
13.	10,00	Fe80:e010:c683:5106:f3e8	Ff02::c	SSDP	208
14.	10,03	Cisco_e1:aa:81	Spanning_tree	STP	60
15.	10,88	Fe80:e010:c683:5106:f3e8	Ff02::1:2	DHCPv6	154
16.	12,03	Cisco_e1:aa:81	Spanning_tree	STP	60
17.	13,30	Cisco_e1:aa:81	CDP/VTP/DTP	DTP	60
18.	13,30	Cisco_e1:aa:81	CDP/VTP/DTP	DTP	90
19.	14,00	Fe80:e010:c683:5106:f3e8	Ff02::c	SSDP	208
20.	14,03	Cisco_e1:aa:81	Spanning_tree	STP	60
21.	16,03	Cisco_e1:aa:81	Spanning_tree	STP	60
22.	17,00	Fe80:e010:c683:5106:f3e8	Ff02::c	SSDP	208
23.	18,03	Cisco_e1:aa:81	Spanning_tree	STP	60
24.	18,67	Cisco_e1:aa:81	Cisco_e1:aa:81	LOOP	60
25.	19,32	Cisco_e1:aa:81	ASUSTEkC_9d:86:8a	ARP	60
26.	19,32	ASUSTEkC_9d:86:8a	Cisco_e1:aa:81	ARP	42
27.	20,00	Fe80:e010:c683:5106:f3e8	Ff02::c	SSDP	208
28.	20,03	Cisco_e1:aa:81	Spanning_tree	STP	60
29.	20,51	40.40.40.3	10.10.10.2	TCP	66
30.	22,03	Cisco_e1:aa:81	Spanning_tree	STP	60
31.	23,51	40.40.40.3	10.10.10.2	TCP	66
32.	24,00	Fe80:e010:c683:5106:f3e8	Ff02::c	SSDP	208
33.	24,03	Cisco_e1:aa:81	Spanning_tree	STP	60
34.	25,22	ASUSTEkC_9d:86:8a	Cisco_00:e3:e1	ARP	42
35.	25,23	Cisco_00:e3:e1	ASUSTEkC_9d:86:8a	ARP	60
36.	26,03	Cisco_e1:aa:81	Spanning_tree	STP	60
37.	27,00	Fe80:e010:c683:5106:f3e8	Ff02::c	SSDP	208
38.	27,50	40.40.40.3	10.10.10.2	SIP/SDP	1095
39.	27,50	10.10.10.2	40.40.40.3	SIP	284
40.	28,30	40.40.40.3	10.10.10.2	SIP	831
41.	30,25	10.10.10.2	40.40.40.3	RTP	214
42.	31,50	40.40.40.3	10.10.10.2	RTP	214
43.	31,75	10.10.10.2	40.40.40.3	RTP	214

 ISBN 978-86-7466-930-3

IV. ZAKLJUČAK

Rezultat jedne realizacije zaštite paketskog telefonskog saobraćaja, predstavljen u ovom radu prikazuje da uspostava i održavanje VPN tunela kao načina zaštite paketskog telefonskog saobraćaja podrazumeva primenu niza protokola, specifično dizajniranih za prenos informacionih sadržaja u realnom vremenu (SIP, SSDP, RTP) kao i protokola za obezbeđenje zaštite u toku prenosa (ISAKMP, IPSec). Specifično za predstavljenu realizaciju predstavlja upotreba linka za prenos različitih servisa i time korišćenje protokola kojima se omogućava konvergencija servisa u prenosu (DTP, SSDP).

LITERATURA

[1] W.Stallings, Osnove bezbednosti mreža: Aplikacije i standardi, Računarski fakultet, Beograd, 2014.
[2] M.Stojanović, V.Aćimović-Raspopović, Savremene IP mreže: Arhitekture, tehnologije i protokoli, Akademska misao, Beograd, 2012.
[3] A.Smiljanić, Osnove i primena Interneta, Elektrotehnički fakultet Univerziteta u Beogradu, Beograd, 2015.
[4] D.Nemec, D.Vukobratović, V.Crnojević, Č.Stefanović, Tehnologija VoIP sistema, Fakultet tehničkih nauka, Novi Sad, 2007.
[5] Security in the Internet Architecture, RFC: 1636, jun 1994, https://datatracker.ietf.org/doc/html/rfc1636
[6] IP Encapsulating Security Payload, RFC: 4303, decembar 2005, https://datatracker.ietf.org/doc/html/rfc4303
[7] Security Architecture for the Internet protocol, RFC: 4301, decembar 2005, https://datatracker.ietf.org/doc/html/rfc4301
[8] Internet Security Association and key Management Protocol, RFC: 2408, novembar 1998, https://datatracker.ietf.org/doc/html/rfc2408
[9] Internet Key Exchange Protocol Version 2 (IKEv2), RFC: 5996, septembar 2010, https://datatracker.ietf.org/doc/html/rfc5996
[10] B.Scheiner, Primenjena kriptografija, Mikro knjiga, Beograd, 2007.
[11] R.Swale, D.Collins, Carrier Grade Voice Over IP, McGraw Hill Professional, 2004.
[12] M.Jevtović, Komunikacioni protokoli Interneta, Akademska misao, Beograd, 2011.
[13] I.Bašičević, "Prilog razvoju arhitekture za obezbeđivanje usluga u računarskim mrežama nove generacije", doktorska disertacija, Fakultet tehničkih nauka, Novi Sad, 2008.
[14] S.Gajin , Principi konfigurisanja računarskih mreža, Akademska misao, Beograd, 2018.

ABSTRACT

The research presented in this paper addresses an example of how to execute packet telephone traffic protection using virtual private network technology through configuring servers for packet telephone traffic and also demonstrates the secure transfer with the use of virtual private network technology in tunnel mode by applying the appropriate protocols for privacy protection, authentication, integrity protection and crypto key exchange. Traffic recording and analysis has been performed using the Wireshark software in both secure and non-secure transfer. The results obtained can help better understand the complex process of setting up a tunnel through the use of simulation software in education.

Packet Telephone Traffic Transfer Protection Using Technology of Virtual Private Networks

Mićo Živanović, Jovan Bajčetić, Ivan Tot

 ISBN 978-86-7466-930-3