STRUČNI RAD – PROFESSIONAL PAPER 10.7251/ZRPIM2201455K

KREIRANJE CLOUD BAZIRANE WEBRTC VOICE OVER IP APLIKACIJE ZA WIRE (ŽIČNE) I WIRELESS KORISNIKE

Alen Kamiš

Visoka škola za uslužni biznis, Cara Lazara bb, 71350 Sokolac, Istočno Sarajevo, Bosna i Hercegovina, alen@vub.edu.ba

SAŽETAK

Ovaj rad predstavlja primjer izrade web aplikacije koja služi za „Voice over IP – VoIP„ pozive unutar i vankompanije. Aplikacija omogućava žičnim i bežičnim korisnicima pristup i upotrebnu „žive“ govorne i video komunikacije zasnovane na webRTC tehnologiji. Kao platformu za razvoj i produkciju odabrani su kontejneri (mikroservisi) zasnovani na Docker platformi koji su implementirani na Microsoft Azure Cloudu.

Ključne riječi: Docker, kontejner, webRTC, Microsoft Azure

UVOD

U vrijeme pandemije, korištenje Voice over IP aplikacija bila je na najvišem nivou. Aplikacije su korištene za mnoge potrebe: školska predavanja, poslovni sastanci, rad kod klijenta i slično. U većini slučajeva aplikacije su bile komercijalnog karaktera. Ovaj rad predstavlja kreiranje Web aplikacije Voice over IP (VoIP), koja koristi open-source webRTC protokol i koja je besplatna. Aplikacija je dizajnirana da koristi više mikroservisa (Docker kontejnera) čime je pokretanje same aplikacije optimizovano i te je isto moguće u veoma kratkom roku. Kao platforma i osnova za mikroservise izabran je Docker. Docker je platforma koja dozvoljava programerima, sistem administratorima i osobama sličnih profesija kreiranje i održavanje aplikacija u softverskim kontejnerima (mikroservisima). Docker je instaliran na Windows 10 radnu stanicu kao razvojnu platformu, a za produkciju odabran je Microsoft Azure Cloud servis.

Docker

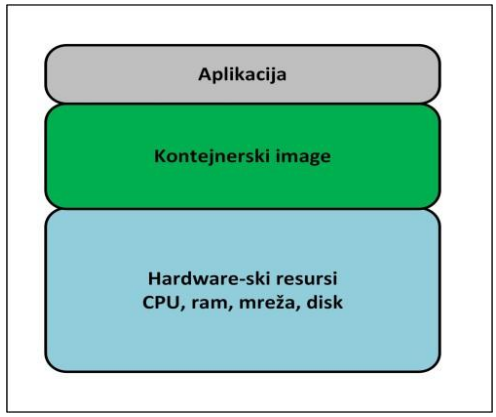
Docker je softver koji je moguće koristiti na Linux i Windows operativnim sistemima. Koristi se kao alat dizajniran za kreiranje, implementaciju i pokretanje aplikacija koristeći kontejnere. Prvenstvena namjena je za programere koji za svoj posao mogu odabrati željenu platformu te na njoj programirati bez da brinu o operativnom sistemu na kojem će njihova aplikacija biti pokrenuta (Docker Softver, 2022). U novije vrijeme sve je više poslovnih aplikacija koje koriste Docker. Prilikom implementacije web aplikacije webRTC korišten je Docker koji je prvobitno podignut na Windows 10 operativnom sistemu te su na ovoj verziji podignuti svi kontejneri. Nakon uspješno implementirane razvojne platforme, kontejneri su prebačeni u Microsoft Azure cloud u svrhu produkcione platforme.

Kontejneri

Kada govorimo o kontejnerima (engl. containers), u kontekstu aplikacija i softvera možemo ih usporediti sa stvarnim kontejnerima za otpremu robe. Prije njihovog izuma i standardizacije slanje robe bio je zahtjevan i skupocjen proces. Ovisno o vrsti robe i metodi slanja bilo je potrebno prilagođavati način pakiranja. Standardizacijom kontejnera izjednačen je način slanja robe neovisno o vrsti, obliku, veličini i načinu prijevoza – brodom, vozom ili kamionom. Količina posla smanjena je uz pojednostavljenje procesa a postignute su i znatne vremenske i cjenovne uštede (Jangla, 2018).

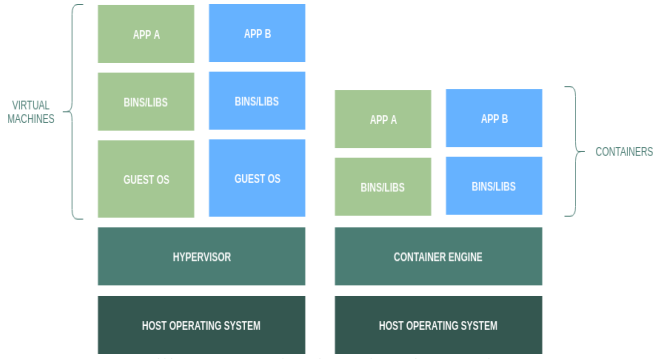
Sličan princip je i u slučaju softverskih kontejnera. U kontejnere postavljamo samo ono što nam je potrebno da bi pokrenuli aplikaciju. Taj kontejner zatim možemo prenositi i pokretati bilo gdje. Umjesto instalacije cijelog operativnog sistema i svog propratnog softvera, softverski

kontejner najčešće sadrži samo sljedeće komponente: samu aplikaciju, potrebne biblioteke, komponente o kojima aplikacija zavise i konfiguracijske datoteke i fascikle (Slika 1). Aplikacija tako postaje neovisna o vrsti distribucije operativnog sistema i infrastrukturi na kojoj istu pokrećemo.



Slika 1. Arhitektura kontejnera.
Figure 1. Container architecture.

Kontejner se vrlo često povezuje ili čak miješa s pojmom virtualna mašina. Virtualne mašine pokreću se unutar programa za virtualizaciju i predstavljaju zapravo nove, zasebne (virtualne) operativne sisteme unutar host-a. Prema tome, imaju svoje biblioteke, programe i sve ostalo te zauzimaju i po nekoliko gigabajta memorijskog prostora. S druge strane, kontejneri su zapravo samo izolirana okruženja, unutar host operativnog sistema (dakle, ne postoji još jedan sloj operativnog sistema), ali sadrže vlastite biblioteke za određene procese i aplikacije, što ih memorijski čini puno manjima u usporedbi sa virtualnim mašinama. Kontejnere se ponekad naziva „vrlo laganim virtualnim mašinama“. U nastavku, na Slici 2, nalazi se skica koja prikazuje glavne tehničke razlike između virtualnih mašina i kontejnera (NetApp Blog, 2018).

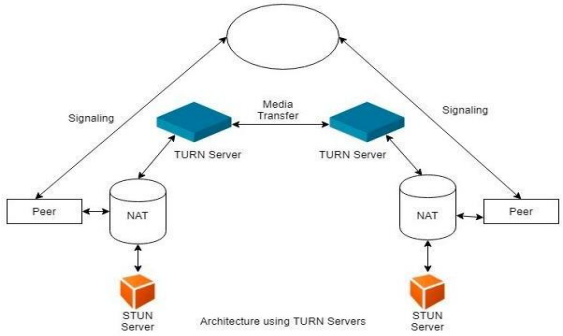


Slika 2. Kontejneri vs virtualne mašine.
Figure 2. Containers vs virtual machines.

WEBRTC WEB APLIKACIJA

WebRTC (Web Real-Time Communication) je besplatan projekat otvorenog koda koji pruža Web pretraživačima i mobilnim aplikacijama komunikaciju u stvarnom vremenu (RTC) putem interfejsa za programiranje aplikacija (API). Isti omogućava rad audio i video komunikacije unutar web stranica omogućavajući tako direktnu međusobnu komunikaciju, te tako eliminirajući potrebu za instaliranjem dodataka ili preuzimanjem izvornih aplikacija. WebRTC podržavaju Apple, Google, Microsoft, Mozilla i Opera preglednici(WebRTC,2022).Razvijena je Web aplikacija koja

omogućava uspostavljanje video poziva koristeći webRTC open-source kod. Arhitekturno, web aplikacija je implementirana koristeći mikroservise i kontejnere koji su hostani na Microsoft Azure. Na sljedećoj slici može se vidjeti arhitektura webRTC aplikacije.



Slika 3. Arhitektura webRTC aplikacije (Agora, 2022).
Figure 3. WebRTC application architecture (Agora, 2022).

Kako radi webRTC aplikacija

WebRTC je open source projekat koji omogućava komunikaciju zvuka, videa i podataka u realnom vremenu u Web i izvornim aplikacijama. U ovom projektu korišten je dio webRTC za zvučnu komunikaciju. WebRTC ima nekoliko JavaScript API-ja od kojih su najpopularniji:

- getUserMedia: omogućeno korištenje inputa (kamera i mikrofon)
- MediaRecorder: snimanje audio i video zapisa
- RTCPeerConnection: stream audio i video između korisnika
- RTCDataChannel: tok podataka između korisnika

WebRTC se može koristiti u web pretraživačima Firefox, Opera i Google Chrome na desktop i mobilnoj varijanti. WebRTC je također dostupan za dodatne aplikacije na iOS i Android uređajima. Da bi webRTC Web aplikacija bila funkcionalna, potreban je signalni „servis“, obično nazvan signalizacija. WebRTC koristi RTCPeerConnection za komunikaciju streaming podataka između pretraživača, ali je također potreban mehanizam za koordinaciju komunikacije i slanje kontrolnih poruka, proces poznat kao signalizacija. WebRTC ne specificira metode i protokole signalizacije. U ovoj implementaciji korišten je Socket.IO za razmjenu poruka, ali postoje i druge alternative. Da bi webRTC aplikacija uspostavila poziv, njeni klijenti moraju razmijeniti sljedeće informacije:

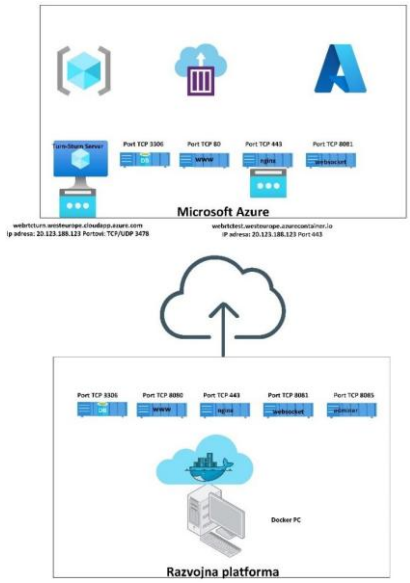
- Poruke kontrole sesije koje se koriste za otvaranje ili zatvaranje komunikacije
- Poruke o grešci
- Metapodatke medija, kao što su kodeci, postavke kodeka, propusni opseg i tipovi medija
- Ključni podaci koji se koriste za uspostavljanje sigurnih veza
- Mrežni podaci, kao što su IP adresa i port domaćina kako ih vidi vanjski svijet

Proces signalizacije zahtijeva mehanizam da klijenti prosljeđuju poruke naprijed-nazad. Taj mehanizam nije implementiran od strane webRTC API-ja, već se mora napraviti samostalno. WebRTC je dizajniran da radi peer-to-peer konekciju, tako da se korisnici mogu povezati najdirektnijim mogućim putem. Međutim, webRTC je napravljen da podržava zadnje mrežne tehnologije u stvarnom svijetu, to jest klijentske aplikacije mogu proći kroz NAT gateway i firewall, ako je u pitanju peer-to-peer uspostava poziva, a ako ne komunikacija prođe direktno. Kao dio ovog procesa, webRTC API koriste STUN servere da dobiju IP adresu korisničkog računara, a TURN serveri da funkcionišu kao relejni serveriu slučaju da međusobna komunikacija ne uspije.

Kriptovanje poziva je obavezna za sve webRTC komponente, a njegovi JavaScript API se mogu koristiti samo u sigurnoj komunikaciji putem https veze.

Dizajn Aplikacije

Na narednoj slici prikazan je šematski prikaz Docker kontejnera na lokalnom računaru i Microsoft Azure platformi.



Slika 4. Prikaz dizajna webRTC aplikacije.
Figure 4. WebRTC application design view.

WebRTC aplikacija je besplatna i može se preuzeti sa cjelokupnim kodom na Github-u (WebRTC-project, 2022).

Aplikacija na Microsoft Azure se sastoji od četiri kontejnera i jedne Linux virtualne mašine. WebRTC aplikacija se sastoji od sljedećih kontejnera:

- Db
- Websocket
- Ngnix
- www

Linux virtualna mašina je podignuta u Microsoft Azure zbog ograničenja na mrežnoj konfiguraciji firewall-a prilikom dozvoljavanja/zabaranjivanja specifičnih portova koji su potrebni za sigurnost sistema. Linux virtualna mašina ima ulogu TURN i STUN servisa.

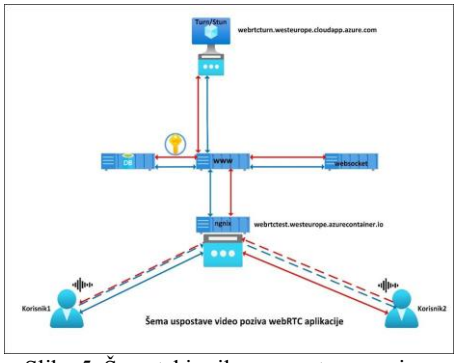
Kontejner db je dio Web aplikacije u kojem je smještena MYSQL baza podataka koja je potrebna za Web site. MYSQL server sadrži baze podatka koje se veoma često koriste kod izrade Web aplikacija. Unutar db kontejnera kreirana je baza koja se zove „web.“ Unutar baze „web“ nalazi se tabela „users“ koja sadrži korisnike koji se registruju na Web aplikaciju.

Kontejner websocket je dio web aplikacije koji radi podršku rada sa websocketima, pošto php programskijezik ne podržava nativno webRTC platformu. Zbog odlične podrške za websockete, za mikroservis razmjene poruka između komponenti iskorišten je node.js, čime su izbjegnuti neželjeni efekti. Konfiguracija za Websocket-e u node.js sastoji se od serverskog i klijentskog dijela. Serverski dio se nalazi u kontejneru „websocket“, a klijentska konfiguracija se nalazi u „www“ kontejneru. Kontejner www je ključan za Web aplikaciju jer u istom se nalazi Apache server. Iz Docker konfiguracije datoteke vidi se da je Web site smješten u /var/www/html folder. Unutar html fascikle nalaze se naredne datoteke neophodne za rad web aplikacije:

- call.php
- functions.php
- index.php
- login.php
- logout.php
- registration.php
- script.js
- socket.io.js

Kontejner ngnix ima ulogu frontend Web servera kojem korisnici prvo pristupaju prilikom pristupanja webRTC aplikaciji. Prilikom pokušaja spajanja, korisnici se povezuju se na ngnix Web server putem tcp porta 443, koji je standard za https sesije. Na samom kontejneru implementiran je i letsencrypt certifikat kako korisnici ne bi naišli na greške prilikom otvaranja aplikacije.

Na sljedećoj slici prikazan je cjelokupni rad webRTC aplikacije od samog pristupanja web stranici pa do uspostave VoIP poziva prema drugom korisniku.



Slika 5. Šematski prikaz uspostave poziva.
Figure 5. Schematic representation of call establishment.

ZAKLJUČCI

U radu je opisan ukratko proces kreiranja webRTC aplikacije. Za Web aplikaciju koja se koristi za video chat putem webRTC protokola korišten je php programski jezik – tehnologija, uz pomoć node.js programskog jezika. webRTC je „open source" platforma i kao takva je veoma interesantna za korištenje. Aplikacija u infrastrukturnom dijelu je napravljena na Docker – kontejner platformi. Docker kontejnerska platforma se pojavila 2013 godine i odmah je postala popularna platforma za razvojna okruženja. Docker kao platforma je sve više korišten i za produkcione sisteme, zbog praktičnosti mikroservisa. Kao platforma ima svoje prednosti u smislu da Web aplikacija brzo starta, podijeljena je u više segmenata i kao takva je manje podložna napadima, a istovremeno su olakšani popravci i servisiranja. Mane Docker platforme vezane su za ograničenja kontejnera, „težu implementaciju", prilagodbu i održavanje (npr. granularni backup/restore aplikacije – osim fizičkog kopiranja).

Na kraju, može se doći do zaključka da će se Docker kontejner platforma u budućnosti dodatno širiti te za očekivati je da u narednim godinama većina aplikacija bude zasnovana na ovoj platformi.

LITERATURA

Agora. (2022). The Past, Present, and Future of WebRTC. Preuzeto 17.03.2022. sa https://www.agora.io/en/blog/past-present- future-of-webrtc/

Jangla, K. (2018). *Accelerating Development Velocity Using Docker: Docker Across Microservices*. Apress.

NetApp Blog. (2018). Containers vs. Virtual Machines (VMs): What's the Difference? Preuzeto 16.03.2022. sa https://www.netapp.com/blog/containers-vs-vms/

WebRTC - Wikipedia. (2022). Report. Preuzeto 17.03.2022. sa https://en.wikipedia.org/wiki/WebRTC
Wikipedia. (2022). Docker softver. Preuzeto 17.03.2022. sa https://bs.wikipedia.org/wiki/Docker_(softver)

CREATING CLOUD BASED WEBRTC VOICE OVER IP APPLICATIONS FOR WIRED AND WIRELESS USERS

Alen Kamiš

The College of Service Business, Cara Lazara bb, 71350 Sokolac, Istočno Sarajevo, Bosna i Hercegovina, alen@vub.edu.ba

ABSTRACT

This document presents a sample of building a Web application which is used for „Voice over IP – VoIP„ calls, internal and external to the company. The application enables wired and wireless users to access "live" voice and video communications based on the webRTC technology. Containers (micro services) were used as a development and production platform based on Docker implemented on Microsoft Azure Cloud.

Keywords: Docker, container, webRTC, Microsoft Azure.