

Fakultet elektrotehnike i računarstva, Zagreb

Raspodjeljeni sustavi  
Laboratorij profila  
Akademska godina: 2017/2018

# Web aplikacija povezana na implementaciju DNS-a ostvarena pametnim ugovorom Ethereum

## Projektna dokumentacija

Voditelji: Federico Matteo Benčić  
Ivana Podnar Žarko

|      |                |            |
|------|----------------|------------|
| Tim: | Toma Majić,    | 0036487650 |
|      | Josip Paladin, | 0036479865 |
|      | Alen Dostal,   | 0036481129 |
|      | Toma Šimunić,  | 0036473333 |

## Sadržaj:

### 1. Opis teme projekta

- 1.1. Pregled problematike
- 1.2. Konačno rješenje

### 2. Tehničke značajke

- 2.1. Blok-lanac
- 2.2. Pametni ugovori
- 2.3. Solidity
- 2.4. Truffle
- 2.5. geth

### 3. Arhitektura sustava

- 3.1. Web3.js
- 3.2. Pametni ugovori

### 4. Upute za korištenje

- 4.1. Korisnik želi razlučiti simboličku adresu
- 4.2. Korisnik želi kupiti simboličku adresu
- 4.3. Korisnik želi otpustiti simboličku adresu prije vremena isteka registracije

### 5. Literatura

## 1. Opis projektnog zadatka

Naš zadatak bio je implementirati sustav za registraciju i razlučivanje simboličkih adresa u IP adrese putem pametnog ugovora Ethereum te implementirati jednostavno web sučelje povezano s pametnim ugovorom putem kojeg je moguće razlučiti/registirati zadanu simboličku adresu.

### 1) Pregled problematike

DNS (Domain Name System) je sustav koji se danas koristi za registraciju novih adresa i za pronalazak IP-adresa prevođenjem simboličkih adresa. Klasični DNS ima nedostatke koji se mogu riješiti uporabom tehnologije blok-lanca i pametnih ugovora. Tehnologija blok-lanca nam rješava problem centraliziranosti DNS-a, a to je jedna točka ispada. Pametni ugovori omogućavaju vlasniku domene da unosi promjene u svojoj domeni.

### 2) Konačno rješenje

Napravili smo jednostavno web-sučelje u kojem korisnik može kupiti,razlučiti ili obrisati domenu. Rješenje nudi jednostavnu implementaciju DNS-a na Ethereum blok-lancu.

## **2. Tehničke značajke**

### **1) Blok-lanac**

Blok-lanac je kontinuirano rastuća lista zabilježbi koje se nazivaju blokovi, a osigurani su i povezani kriptografskim metodama. Svaki blok se sastoji od hash-pokazivačana prethodni blok, vremenske oznake i podataka transakcija. Blok-lanac je po dizajnu otporan na promjenu podataka. Za uporabu kao distribuirani zapisnik blok-lanac koristi peer-2-peer zajedničkim protokolom potvrđuje ispravnost bloka. Naše programsko rješenje se odvija na Ethereum blok-lancu

### **2) Pametni ugovori**

Pametni ugovor je računalni ugovor kojem je namjena digitalno olakšati, verificirati ili izvršiti ugovor. Omogućavaju izvršavanje transakcija bez posrednika. Svrha im je pružiti sigurnost veću od one koju pružaju tradicionalno ugovorno pravo i smanjiti cijenu transakcija ugovora. U digitalnom obliku je to omogućeno razvojem "Byzantine fault tolerant" algoritama koje koristi i Ethereum.

### **3) Solidity**

Solidity je programski jezik usmjeren ka pisanju i implementaciji pametnih ugovora i koristi se u raznim blok-lanac platformama. Programi pisani u Solidityu se pokreću na EVM-u (Ethereum Virtual Machine). Solidity omogućava pisanje aplikacija koje implementiraju samoizvršavajuću poslovnu logiku ugrađenu u pametne ugovore.

#### 4) Truffle

Truffle je razvojna okolina i testni okvir za Ethereum. Truffle sadrži kompilacije ugrađenih pametnih ugovora, automatsko testiranje ugovora za ubrzani razvoj, upravljanje mrežama, konzola za direktnu komunikaciju s ugovorom i izvršitelj eksternih skripti.

#### 5) Geth

Geth je sučelje komandne linije za pokretanje cijelog Ethereum čvora implementirano u jeziku Go. Pokretanjem Getha se može sudjelovati u "Ethereum frontier live network" te rudariti Ethereum, vršiti transakcije, stvarati ugovore i istraživati stanja blok-lanca.

### 3. Arhitektura sustava

Web aplikacija EthereumDNS je decentralizirana aplikacija na Ethereum block-lancu. Na Ethereumu se nalazi pametni ugovor koji predstavlja logiku aplikacije - provodi dodjelu, razlučivanje, te brisanje domena na zahtjev korisnika. Pametni ugovor napisan je u programskom jeziku Solidity. Ugovor je objavljen na Ethereum test mrežu - Ropsten. Povezivanje s Ropsten mrežom obavljeno je pomoću Google Chrome "plug-ina" MetaMask koji omogućava povezivanje s mrežom preko udaljenog čvora.

Korisničko sučelje izrađeno je pomoću HTML, CSS i JavaScript tehnologije, a s logikom je povezano preko Web3.js API-ja.

#### 1. Web3.js

Web3.js je sučelje koje pruža objekte i metode potrebne za komunikaciju s mrežom. Komunikacija s čvorom (lokalnim ili udaljenim preko MetaMask-a) odvija se preko RPC protokola.

```
1  $(document).ready(() => {
2      if(typeof web3 !== undefined) {
3          window.web3 = new Web3(web3.currentProvider);
4      } else {
5          console.log('No injected web3 found');
6
7          var provider = 'http://localhost:8545';
8          window.web3 = new Web3(new Web3.providers.HttpProvider(provider));
9      }
10
11     console.log('Web3 provider: ', window.web3.currentProvider);
12 });
```

Slika 1 Generiranje web3 objekta

Prilikom učitavanja aplikacije provjerava se postoji li definirani web3 “provider” (MetaMask), te ako postoji kreira se web3 objekt s njegovim parametrima. U suprotnom se zadaje lokalni “provider”, npr localhost:8545 u slučaju da je pokrenut lokalni čvor na npr. geth klijentu (Slika 1).

Jedan od važnijih objekata kojeg sadrži web3 je web3.eth objekt. On pruža brojne metode koje pomažu pri interakciji s Ethereum blok-lancem i pametnim ugovorima.

```
101 var abiDefinition = JSON.parse(abi);  
102 var contract = web3.eth.contract(abiDefinition);
```

Slika 2 Primjer korištenja eth

Jedan od primjera korištenja web3.eth objekta je dohvaćanje pametnog ugovora s blok-lanca. Da bi se pametni ugovori mogli koristiti unutar aplikacije potrebno ih je dohvatiti te kreirati novi objekt koji sadrži sve njegove metode. Objekt se stvara pomoću “abi” (Application Binary Interface) koji se dobije u json formatu (Slika 2).

Pozivanje metoda pametnog ugovora obavlja se naredbama call() ili sendTransaction() ovisno o tome radi li se o izmjeni podataka ili samo dohvaćanju. U slučaju dohvaćanja podataka dovoljno je nad metodom pozvati .call() naredbu zato što ne radi nikakve promjene na podacima. U slučaju da dolazi do promjene koristi se metoda sendTransaction(), zato što svaka nova promjena mora biti spremljena na blok-lancu.

```

dnsContract.resolveDomain.call(domain, (err, res) => {
  if(err) {
    var imgSrc = './images/error.png';

    var html = '<h1>Pogreška</h1>'
      + '<p>Tražena domena ne postoji!</p>'
      + '<div class="error-img"><img src=' + imgSrc + ' height="175" width="175"></div>';

    _Modals.openModal();
    $('.modal').fadeIn(200).append(html);
  } else {
    $('.domain-resolve').val(res);
  }
});

```

Slika 2.2 Dohvaćanje podataka s blok-lanca ( call() )

```

dnsContract.buyDomain.sendTransaction(domain, ipAddr, trxObject, (err, res) => {
  if(err) {
    console.log('Error: ', err);
  } else {
    console.log(res);

    var imgSrc = './images/success.png'
    var html = '<img src=' + imgSrc + ' height="175" width="175"></div>'
      + '<p>Uspješno izvršena transakcija</p>';

    $('.modal').fadeIn(200).empty();
    $('.modal').fadeIn(200).append(html);
  }
});

```

Slika 2.2 Izmjena podataka na blok-lancu ( sendTransaction() )



## 2. Pametni ugovor

Pametni ugovor predstavlja “back-end” sustava na kojem su sadržani podaci bitni za aplikaciju (domene, njihove IP adrese i vlasnici) te sadrži javno sučelje koje omogućuje izmjenu tih podataka. Također sadrži poslovnu logiku u obliku definiranih cijena za kupovinu domene te pravila po kojima razlučujemo domene na zahtjev.

Za razvoj, testiranje i objavu pametnih ugovora na Ropsten korišteni su remix, te truffle razvojni okvir koji nudi prevoditelj za Solidity (naredba ‘truffle compile’). Prilikom objave pametnog ugovora potrebno je pokrenuti migraciju koja objavljuje ugovor na mrežu objavom ‘truffle migrate --network ropsten’.

```
1 |var EthereumDNS = artifacts.require('./EthereumDNS.sol');
2 |
3 |module.exports = function(deployer) {
4 |    deployer.deploy(EthereumDNS);
5 |}
```

Slika 2.1 Objava EthereumDNS ugovora

Prilikom obavljanja migracije koristi se ‘deployer’ kako bi se pametni ugovor objavio na ropsten. Ta migracija stvara novu transakciju, koja se potvrđuje s povratnom adresom pametnog ugovora, nakon što je transakcija spremljena unutar bloka na blok-lancu.

Slika 3 prikazuje osnovni model podataka kojim opisujemo domenu i mapu u koju spremamo trenutno stanje aplikacije (vlasnike i njihove pripadajuće domene).

```

11 // Define struct for reserving domain
12 struct DNSRecord {
13     // Ethereum address for owner of the domain
14     address owner_addr;
15     // IP address attached to domain
16     string ip_addr;
17     // Time remaining before record is not valid
18     uint256 end_timestamp;
19 }
20
21 // Define current price for a day in wei
22 uint private CurrentPrice = 1;
23
24 // List containing domains (string) and their owners
25 mapping (string => DNSRecord) dns_records;

```

Slika 3 Osnovni model aplikacije

Javno sučelje pametnog ugovora sastoji se od sljedeće 4 metode kojima mijenjamo ili dohvaćamo trenutno stanje aplikacije. Metoda “resolveDomain” kao parametar prima ime domene za koju korisnik želi saznati IP adresu. Pozivom ove metode pametni ugovor dohvaća iz mape svih domena onu koja odgovara parametru i (ukoliko postoji i nije istekla) je vraća korisniku (Slika 4).

```

30 // Resolves IP address based on domain name
31 function resolveDomain(string domain) public constant returns(string) {
32     var dnsRecord = dns_records[domain];
33     if(dnsRecord.isDefined && dnsRecord.end_timestamp > now){
34         return dnsRecord.ip_addr;
35     }else{
36         // address does not exists or registration time was expired
37         throw;
38     }
39 }

```

Slika 4 Dohvaćanje IP adrese za domenu

Metoda “buyDomain” (Slika 5) korisniku omogućuje da na određeno vrijeme zakupi domenu i pridoda joj željenu IP adresu. Za to plaća iznos u Etheru koji je također definiran unutar ugovora i pohranjen u varijabli “CurrentPrice”

```

42 ▾ function buyDomain(string domain, string ip_addr) public payable {
43 ▾     if(dns_records[domain].isDefined && dns_records[domain].end_timestamp > now){
44         // address exists and registration time was not expired
45         throw;
46     }
47 ▾     else{
48         var end_timestamp = calculateTimeForAmount(msg.value) + now;
49         dns_records[domain] = DNSRecord(msg.sender, ip_addr, end_timestamp, true);
50     }
51 }
52

```

Slika 5 Zakup domene

Metodom “deleteDomain” vlasniku domene je omogućeno da, uz naknadu, otpusti domenu i time prestaje biti njen vlasnik te je domena dostupna drugim korisnicima (Slika 6).

```

53 // Delete domain if you are an owner of the domain
54 ▾ function deleteDomain(string domain) public {
55     if(dns_records[domain].owner_addr == msg.sender && dns_records[domain].isDefined == true){
56         dns_records[domain].isDefined = false;
57     }
58 ▾     else{
59         throw;
60     }
61 }

```

Slika 6 Otpuštanje domene

Metode “getCurrentPrice” i “updateCurrentPrice” (Slika 7) služe za dohvat trenutno definirane cijene zakupa domene na vrijeme od jednog dana te, po potrebi, njeno mijenjanje. Mijenjanje trenutne cijene dozvoljeno je samo vlasniku pametnog ugovora.

```

63 // Update current price if you are an owner of the contract
64 ▾ function updateCurrentPrice(uint newPrice) public ifOwner{
65     CurrentPrice = newPrice;
66 }
67
68 // Get current price for a day
69 ▾ function getCurrentPrice() public constant returns(uint){
70     return CurrentPrice;
71 }

```

Slika 7 Dohvat i mijenjanje trenutne cijene

Pametni ugovor sadrži i privatne varijable, metode i modifikatore koji pomažu pri izračunu cijene koju korisnik mora platiti za domenu ali i provjeru kojom utvrđujemo je li pozivatelj metode ujedno u vlasnik ugovora (u slučaju promjene cijene). Vlasnik ugovora zapisan je u privanoj varijabli “contract\_owner” koja dobije vrijednost samo jednom, i to pri postavljanju ugovora na mrežu (Slika 9).

```
73 // Fail safe for hackers - check amount against our current price
74 function calculateTimeForAmount(uint amount) private returns (uint256){
75     // Calculate number of days (no decimals)
76     var numberOfDays = amount / CurrentPrice;
77     // Return number of seconds
78     return numberOfDays * 86400;
79 }
80
81 // Modifier for determining whether function caller is contract owner
82 modifier ifOwner(){
83     if(msg.sender == contract_owner){
84         _;
85     }else{
86         throw;
87     }
88 }
```

Slika 8 Privatne metode i modifikatori

```
4 // Contract owner address
5 address contract_owner;
6
7 // Constructor for storing contract owner address
8 function EthereumDNS() public {
9     contract_owner = msg.sender;
10 }
11
```

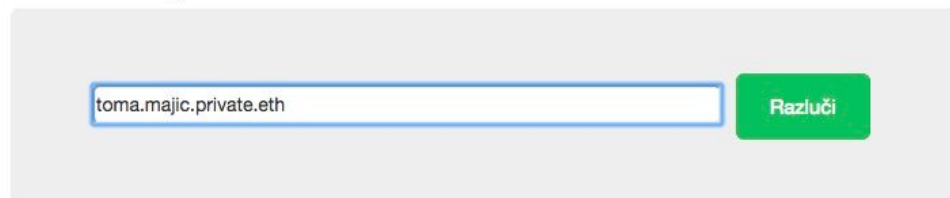
Slika 9 Postavljanje vlasnika ugovora

## 4. Upute za korištenje

### 4.1. Korisnik želi razlučiti simboličku adresu

4.1.1 Adresa postoji u sustavu i vrijeme registracije nije isteklo  
Korisnik dolazi na web stranicu te u odgovarajuće polje upisuje simboličku adresu koju želi razlučiti. Pritiskom na gumb “Razluči” sustav u istom polju ispisuje željenu IP adresu.

#### Razlučivanje domene:



A screenshot of a web interface for domain resolution. It features a light gray rectangular box. Inside the box, on the left, is a text input field with a blue border containing the text "toma.majic.private.eth". To the right of the input field is a green rectangular button with the white text "Razluči".

Slika 4.1.1 Unos domene

#### Razlučivanje domene:



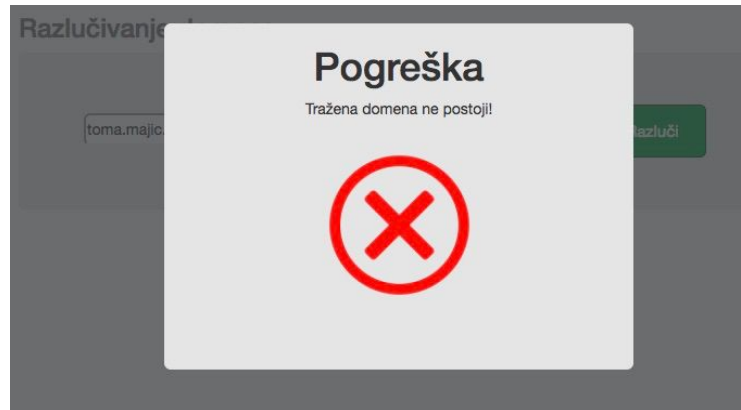
A screenshot of the same web interface as in the previous image. The text input field now contains the IP address "111.111.111.1". The green "Razluči" button remains to the right of the input field.

Slika 4.1.2 Razlučena domena

4.1.2 Adresa postoji u sustavu, ali vrijeme registracije je isteklo  
Korisnik dolazi na web stranicu te u odgovarajuće polje upisuje simboličku adresu koju želi razlučiti. Pritiskom na gumb “Razluči” dolazi do greške jer sustav nije u mogućnosti razlučiti željenu IP adresu jer je vrijeme registracije te adrese isteklo.

#### 4.1.3 Adresa ne postoji u sustavu

Korisnik dolazi na web stranicu te u odgovarajuće polje upisuje simboličku adresu koju želi razlučiti. Pritiskom na gumb “Razluči” dolazi do greške jer u sustavu ne postoji registrirana željenu IP adresa.

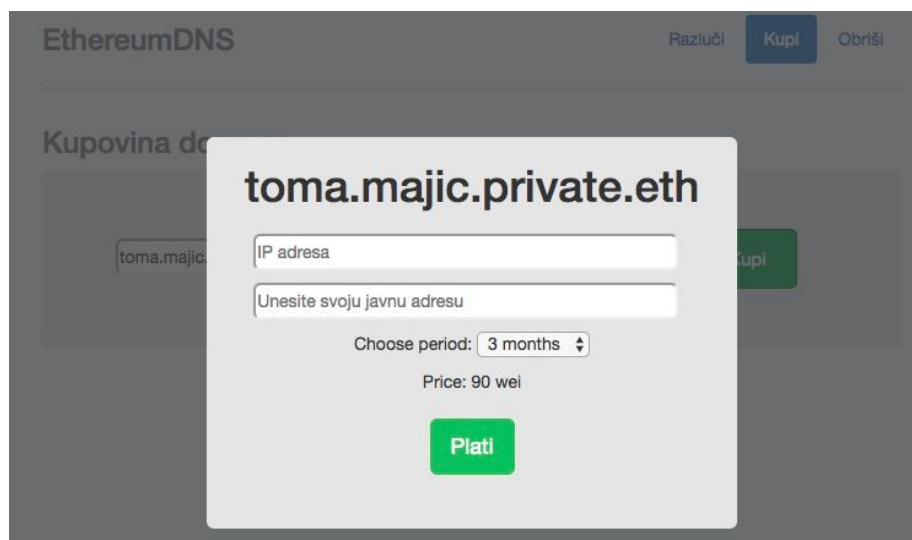


Slika 4.1.3 Domena ne postoji

#### 4.2. Korisnik želi kupiti simboličku adresu

##### 4.2.1 Adresa je slobodna

Korisnik dolazi na web stranicu te odabire karticu za kupnju simboličke adrese. Korisnik u odgovarajuće polje upisuje simboličku adresu koju želi kupiti. Pritiskom na gumb “Kupi” sustav prikazuje modal u kojeg je potrebno upisati odgovarajuću IP adresu, upisati broj terećenog računa te iz padajućeg izbornika odabrati vrijeme trajanja registracije. Ispod padajućeg izbornika ispisuje se cijena registracije. Kada je korisnik popunio sve podatke odabire gumb “Plati” te sustav obavlja transakciju i registrira željenu adresu.



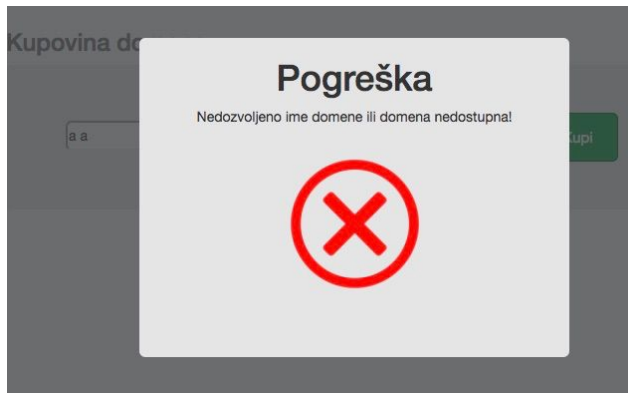
Slika 4.1.1 Modal za kupnju

#### 4.2.2 Adresa je zauzeta ali je vrijeme registracije isteklo

Korisnik dolazi na web stranicu te odabire karticu za kupnju simboličke adrese. Korisnik u odgovarajuće polje upisuje simboličku adresu koju želi kupiti. Pritiskom na gumb “Kupi” sustav prikazuje modal u kojeg je potrebno upisati odgovarajuću IP adresu, upisati broj terećenog računa te iz padajućeg izbornika odabrati vrijeme trajanja registracije. Ispod padajućeg izbornika ispisuje se cijena registracije. Kada je korisnik popunio sve podatke odabire gumb “Plati” te sustav obavlja transakciju. Adresa se poništava prijašnjem vlasniku i pridodjeljuje korisniku kao novom vlasniku.

#### 4.2.3 Adresa je zauzeta i vrijeme registracije nije isteklo

Transakcija koju korisniku obavlja neuspješno će se izvršiti, jer je adresa već dodijeljena drugom korisniku.



Slika 4.2.2 Pogrešna domena

#### 4.3. Korisnik želi otpustiti simboličku adresu prije vremena isteka registracije

Korisniku se prikazuje modal za potvrdu brisanja domene. Korisniku je u modalu objašnjeno da će ga brisanje domene koštati određenu naknadu, te da za potvrdu mora unijeti svoju adresu. Ukoliko korisnik zaista želi obrisati svoju domenu, on unosi svoju adresu te prihvaća uvjete pritiskom na gumb “Obriši”.



Slika 4.3.1 Modal za brisanje



#### 4.3.1 Korisnik je vlasnik adrese

Ako je korisnik vlasnik domene koju pokušava otpisati, transakcija će se uspješno izvršiti, te on više neće biti vlasnik domene.

#### 4.3.2 Korisnik nije vlasnik adrese

U suprotnom, ako korisnik nije vlasnik domene, transakcija neće biti izvršena i doći će do pogreške

## 5. Literatura

Geth - <https://github.com/ethereum/go-ethereum/wiki/geth>

Truffle - <http://truffleframework.com/docs/>

Blok-lanac - <https://en.wikipedia.org/wiki/Blockchain>

Solidity - <https://solidity.readthedocs.io/en/develop/index.html>

Byzantine fault tolerance -  
[https://en.wikipedia.org/wiki/Byzantine\\_fault\\_tolerance](https://en.wikipedia.org/wiki/Byzantine_fault_tolerance)

Pametni ugovori - [https://en.wikipedia.org/wiki/Smart\\_contract](https://en.wikipedia.org/wiki/Smart_contract)