Tomcat 9.x / Linux: Zaštita aplikacija i podataka

pet, 2019-01-04 13:51 - Ratko Žižek

Instalirali smo Javu i Tomcat na CentOS, odradili osnovne provjere funkcionalnosti (vidi: https://sysportal.carnet.hr/node/1834 [1])... sve je ok. Vrijeme je da razmislimo kako ćemo zaštititi našeg Mačka od napada s mreže.

Tomcat uživa status softverskog uratka s vrlo malo evidentiranih ranjivosti, još tijekom instalacije primijenili smo neke bazične mjere zaštite (npr., primjena tar.gz paketa i servisnog računa) pa sada, ako smo mu namijenili neki posao u lokalnoj mreži, možemo odmah prijeći na prihvat aplikacija. S druge strane, opslužuje li naš web server klijente koji dolaze s nesigurnih mreža, bilo direktno ili indirektno (kroz reverzni proxy), imamo dodatnog posla, posebice ako su podaci koji cirkuliraju na relaciji klijent <-> Tomcat klasificirani nekim internim ili zakonskim normama.

Sigurnosno zbrinjavanje Tomcata svakako trebamo započeti primjenom smjernica izloženih na adresi http://tomcat.apache.org/tomcat-9.0-doc/security-howto.html [2], većina ih je razumljiva i lako provediva. Mi ćemo se fokusirati na varljivo jasne savjete i primjere, naime, tijekom njihove implementacije pojave se razne dileme ili stvar jednostavno ne radi. Guglanje pomaže, svakako, ali samo ako smo voljni studirati prikazanu građu - pažljivo čitati, uspoređivati i isprobavati savjete jer ima ih svakojakih. Kakogod, vaš autor se prihvatio tog posla, slijedi provjereno štivo na temu zaštite Tomcata 9.x.

1. Zbrku kontradiktornih savjeta glede možebitnih napada kroz port TCP 8005 na kojem Tomcat očekuje naredbu za prestanak rada, i kako tome doskočiti, rasplest ćemo nižim smjernicama:

- nije potrebno mijenjati defaultni port - on prihvaća konekcije samo s loopback adaptera, iznutra; također, redovito je zatvoren vanjskom svijetu lokalnim ili mrežnim vatrozidom;
- potrebno je promijeniti izraz SHUTDOWN u nešto osobito jer taj izraz je u stvari lozinka a bez nje niti jedan user-mode proces ne može isključiti Tomcat kroz port 8005;
- u specifičnim situacijama, recimo da Tomcat opslužuje klijente s ovećim pravima neophodnim za razvoj aplikacija, možemo i onemogućiti taj port u conf/server.xml: <Server port="-1" shutdown="L0z1nka">. Sada Tomcat može spustiti samo superuser naredbom *kill*, uočite, to je još uvijek regularni način stopiranja procesa.

2. Security Lifecycle Listener, u conf/server.xml predstavljen elementom <Listener className="org.apache.catalina.security.SecurityListener"/>, onemogućen je kako ne bi ometao inicijalno postavljanje Tomcata. Naime, on sprječava podizanje web servera ukoliko nisu zadovoljeni određeni sigurnosni preduvjeti jer tada, po mišljenju ASF-a, njihov produkt postaje isuviše nezaštićen. Nažalost, niti nakon višednevnog rovarenja po Webu nisam pronašao pregled svih provjera koje odrađuje, no to nije razlog da ga ignoriramo. Znači, kad obavimo osnovnu konfiguraciju Tomcata, uključit ćemo ga, restart, pa ako se Tomcat ne digne zavirit ćemo u conf/catalina.out da vidimo što nam zamjera.

3. Kad jednom upogonimo alate Manager i Host Manager, u conf/tomcat-users.xml imat ćemo lozinke admina Tomcata u čitljivom obliku. Neugodna situacija ali, srećom, riješiva jer su nam ASF-vci omogućili digestiranje tih lozinki uporabom algoritama klase SHA-2. Upareno s dodatnim

Tomcatovim kontrolama, rješenje zadovoljava i high-security normu ukoliko su lozinke oblikovane "po pravilima službe" (ukratko, ekstradugačke i kompleksne). U primjeru što slijedi provući ćemo lozinku *L0zin-ka:* admina Tomica kroz SHA-512 hashing funkciju (uključeno je soljenje hasha), usput se suprotstaviti brute force napadima probijanja lozinke.

a) Izgled conf/server.xml nakon podešavanja defaultnog virtualnog hosta za sprječavanje brute force napada i za digestiranje lozinki korisnika upisanih u conf/tomcat-users.xml.

```
<Engine name="Catalina" defaultHost="localhost">

<Realm className="org.apache.catalina.realm.LockOutRealm" failureCount="3"
lockoutTime="1800">

<Realm
className="org.apache.catalina.realm.UserDatabaseRealm" resourceName="UserDatabase">

<CredentialHandler
className="org.apache.catalina.realm.MessageDigestCredentialHandler" algorithm="SHA-512" />

</Realm>

</Realm>
```

b) U komandnoj ljusci, u /bin direktoriju, zadajemo:

```
./digest.sh -a SHA-512 -h org.apache.catalina.realm.MessageDigestCredentialHandler L0z1n-ka:
```

Naredba će prikazati digestiranu lozinku, taj podulji niz kopiramo u conf/tomcat-users.xml (vidi naredni korak)

c) Izgled tomcat-users.xml nakon zamjene Tomičine tekstualne lozinke digestiranom:

```
<role rolename="admin-gui"/>

<role rolename="admin-script"/>

<role rolename="manager-gui"/>

<user username="tomica" password="47744d3e8a...(itd.)...33dcc94" roles="admin-gui,manager-
gui"/>

<user username="tadmin1" password="Naze1e-Nad01i.." roles="admin-script"/>

<user username="tadmin2" password="Mual1:Schep0n1a" roles="manager-gui"/>
```

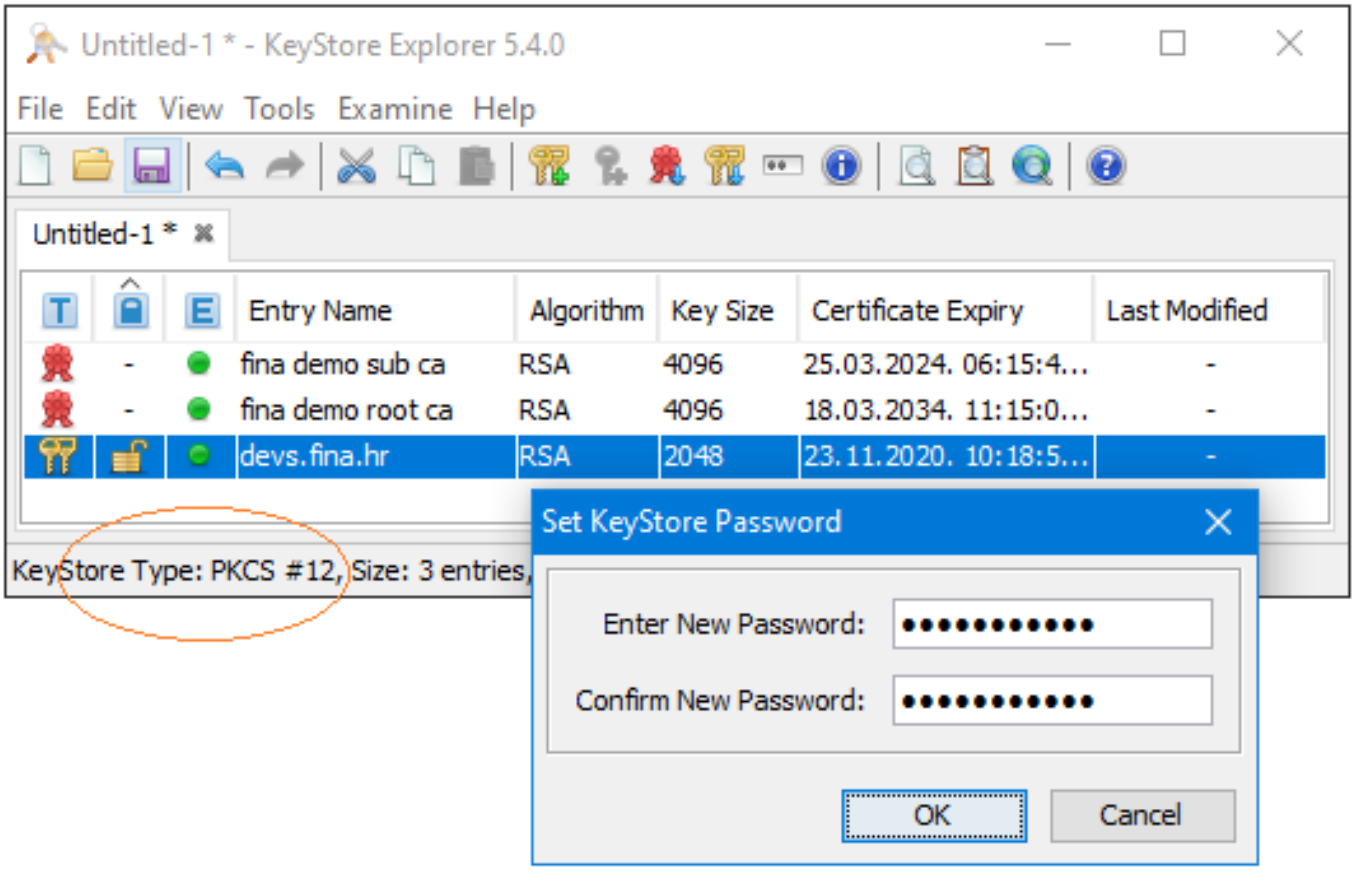
Nadalje Tomica tijekom ulogiravanja upisuje lozinku L0z1n-ka:, kao i ranije. Pogriješi li više od triput, bit će mu onemogućen pristup daljnjih pola sata; događaj se bilježi u conf/catalina.out pa možemo definirati alert.

4. Tuneliranje i enkripcija HTTP prometa danas je podrazumijevani oblik računalne komunikacije. TLS nam treba i za komercijalne aplikacije i za Tomcatove "webolike" admin alate. Pravilo je, naime, da se u okolinama visoke razine zaštite podataka i samog sustava TLS primjenjuje i na relaciji proxy -> Tomcat, znači, Mačak zaprima konekcije na port TCP 8443. Zaštita HTTP sesija ka Manageru i Host Manageru, složit ćemo se, neupitna je potreba.

Prvo ćemo kreirati keystore, datoteku-skladište digitalnih certifikata. Za potrebe Tomcata koristimo Finin demo (besplatni) certifikat imena devs.fina.hr u .pfx formatu. Certifikati CA-eva Finine Demo

PKI infrastrukture javno su dostupni - moraju biti - pa smo i njih skinuli, u .cer formatu, na admin stanicu. Nećemo se zamarati vrijednim ali zahtjevnim Openssl i Keytool alatima; na svoju stanicu - svejedno je li pod Windows ili Linux OS-om - instalirali smo besplatan i susretljiv KeyStore Explorer. Kreirat ćemo PKCS #12 keystore jer je napredniji od .jks formata.

- Create a new KeyStore > odabrati format PKSC #12
- Tools > Import trusted certificate > importirati certifikat Fininog root CA
- Tools > Import trusted certificate > importirati certifikat Fininog issuing CA
- Tools > Import key pair > PKCS #12 > importirati TLS certifikat (plus lozinka kojom štitimo privatni ključ) > postaviti lozinku
- pokrenuti proceduru spremanja, ime neka bude devs-kstore i postaviti lozinku kojom štitimo tu datoteku (na slici)
- kopirati devs-kstore na CentOS u /opt/tomcat9/tomcerts/ + primijeniti na tomcerts chown -R u korist računa tomsvc.



Slijedi pipaviji posao - osposobljavanje Tomcatovog defaultnog hosta "localhost" za prihvat HTTPS konekcija. Napravimo si rezervnu kopiju datoteke conf/server.xml, potom u editoru poput Nano otvorimo izvornu server.xml i krenemo s modifikacijama.

a) Pametni kakvi jesmo, već smo shvatili da smo samo zaboravna ljudska bića, utoliko, prvi zahvat je u stvari podsjetnik i za nas i za naše kolege; ispod elementa <Host name="localhost" appBase="webapps" unpackWARs="true" autoDeploy="true" definiramo mrežno ime defaultnog virtualnog hosta.

<Alias>devs.fina.hr</Alias>

.... kod ....

</Host>

b) Zakomentiramo ili izbrišemo postojeći konektor za HTTPS konekcije, prepoznatljiv je po atributu *<Connector port="8443*", i umjesto njega ukopiramo niži tekst, to je HTTPS konektor konfiguriran za TLS 1.2 konekcije. Svakako pobrišemo oble zagrade i sve što je u njima jer to su pojašnjenja, dio su ovog članka a ne konektora.

<Connector port="8443" protocol="org.apache.coyote.http11.Http11NioProtocol"

maxThreads="400" (atribut se redovito navodi jer služi za fino ugađanje Tomcata)

enableLookups="false" (doprinosi performansama)

SSLEnabled="true" (obavezno za TLS konekcije)

scheme="https" (obavezno za TLS konekcije)

secure="true" (specifičan atribut, vidi dokumentaciju HTTP konektora)

sslProtocol="TLSv1.2" (namećemo zadnju općeprihvaćenu verziju TLS protokola)

sslEnabledProtocols="TLSv1.2" (namećemo zadnju općeprihvaćenu verziju TLS protokola)

honorCipherOrder="true" (prepreka napadu naguravanja ranjivih algoritama)

ciphers="TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384,TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384,

TLS_ECDH_RSA_WITH_AES_256_GCM_SHA384,TLS_ECDH_ECDSA_WITH_AES_256_GCM_SHA384,

TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256,TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256,

TLS_ECDH_RSA_WITH_AES_128_GCM_SHA256,TLS_ECDH_ECDSA_WITH_AES_128_GCM_SHA256,

TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384,TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384,

TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA,TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA,

TLS_ECDH_RSA_WITH_AES_256_CBC_SHA384,TLS_ECDH_ECDSA_WITH_AES_256_CBC_SHA384,

TLS_ECDH_RSA_WITH_AES_256_CBC_SHA,TLS_ECDH_ECDSA_WITH_AES_256_CBC_SHA,

TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256,TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256,

TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA,TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA,

TLS_ECDH_RSA_WITH_AES_128_CBC_SHA256,TLS_ECDH_ECDSA_WITH_AES_128_CBC_SHA256,

TLS_ECDH_RSA_WITH_AES_128_CBC_SHA,TLS_ECDH_ECDSA_WITH_AES_128_CBC_SHA"

keystoreType="PKCS12" (default je JKS pa ovo moramo navesti)

keystoreFile="/opt/tomcat9/tomcerts/devs-kstore"

keystorePass="Zap0rka!" />

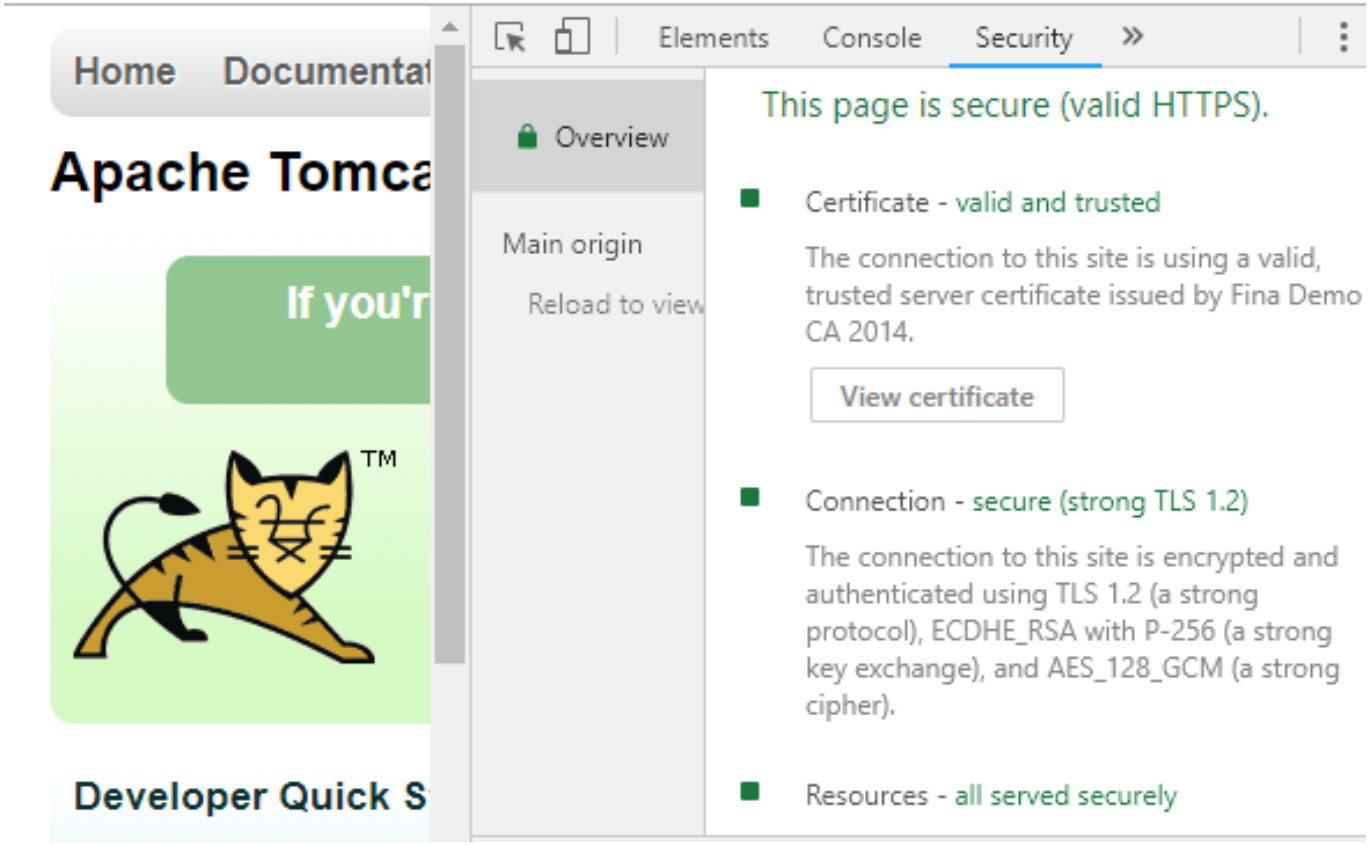
Atribut Ciphers (specificira algoritme zaštite TLS konekcije i podataka u transportu) neozbiljno je popratiti kratkim komentarom, nije na odmet upoznati se s ovim štivom: https://www.owasp.org/index.php/TLS_Cipher_String_Cheat_Sheet [3]. Usput, upravo smo se OWASP preporuka držali pri odabiru ciphersa za našu konfiguraciju.

c) Nakon restarta Tomcata slijedi probno spajanje s admin stanice kao klijentskog računala (otvorili smo port TCP 8443 na vatrozidu servera, jel' tako?). U lokalno skladište certifikata instaliramo spomenuta dva Finina CA certifikata a u lokalnu Hosts unesemo IP adresu Tomcat servera i vežemo ju uz ime devs.fina.hr. Za uobičajeno spajanje na glavnu stranicu u preglednik ćemo upisati https://devs.fina.hr:8443.

d) Ako nam nisu potrebni, sada možemo onesposobiti HTTP i AJP konektore. Doduše, komuniciramo li direktno sa Tomcatom, možda ćemo htjeti postaviti skretanje svih HTTP zahtjeva ka svim aplikacijama (tipično: http://devs.fina.hr:8080) na HTTPS, postavljanjem filtera u conf/web.xml, neposredno iznad elementa </web-app>. Da bi ova redirekcija radila, HTTP konektor mora biti aktivan!

```
<!-- = redirekcija s HTTP na HTTPS = -->

<security-constraint>

  <web-resource-collection>

  <web-resource-name>HTTP-to-HTTPS</web-resource-name>

  <url-pattern>/*</url-pattern>

  </web-resource-collection>

  <user-data-constraint>

  <transport-guarantee>CONFIDENTIAL</transport-guarantee>

  </user-data-constraint>

</security-constraint>
```

Niža slika nam pokazuje da smo dobro zbrinuli Tomcat i njegove aplikacije.



5. Tragikomična zbrka vlada oko kontrole pristupa admin aplikacijama Manager i Host Manager po IP adresi, pa ćemo i to obraditi. U nižem primjeru koristimo jedan od Tomcatovih ventila koji ispituju ili modificiraju HTTP-request pakete prije nego što ih na obradu preuzme Tomcatova mašinerija. Prema admin aplikacijama dopustit ćemo propuštanje samo onih paketa koji dolaze s, recimo, dvije IP adrese u vlasništvu dviju admin stanica. Djelujemo na datoteke /META-INF/context.xml aplikacija Manager i Host Manager tako da u njih upišemo kako je niže prikazano. Sintaksa dopušta uporabu asteriska, recimo, izrazom 192.168.10.* dozvoljavamo pristup svim hostovima s predmetne interne mreže.

```
<Valve className="org.apache.catalina.valves.RemoteAddrValve"
allow="192.168.10.15|192.168.10.20" />
```

6. Niže mjere zaštite ne možemo primijeniti bez dobre suradnje s aplikativcima, utoliko, skrećemo na njih pozornost ali ih ne razrađujemo jer su priča za sebe:

- Aktiviranje Security Managera - nametanjem pravila aplikacijama (u ovu kategoriju spada i Tomcat jer je i on Java aplikacija) posredstvom datoteke conf/catalina.policy, kojim će aplikativnim i sistemskim resursima pristupati, i na koji način, efektivno postavljamo aplikacije u sandboxing režim rada. ASF upozorava da će pokretanje Tomcata s opcijom -security (catalina.sh start -security ili startup.sh - security) vrlo vjerojatno izazvati nepravilan rad aplikacija ukoliko iste nisu razvijane za rad pod kontrolom SM-a. Ovo djeluje obeshrabrujuće jer sugerira dosta posla no, s druge strane, i dobitak je velik - Security Manager značajno unaprijeđuje zaštitu sustava kao cjeline. Napomena: Security Lifecycle Listener iz 2. točke nije povezan sa SM-om.
- Odabir Security Realm komponente - čime se, u suštini, s Tomcatom povezuje izvorište korisničkih računa i aplikativnih uloga za jednu ili više aplikacija. Realm je višeslojna softverska konstrukcija koja Tomcatu omogućuje zaštitu aplikacija (uključujući sebe) i pripadajućih joj podataka tako što provjerava identitet korisnika i njegove dozvole. U produkcijskim implementacijama tehnološka osnovica realma redovito je neka baza podataka ili LDAP servis, tada se bavimo JDBC/DataSource ili JNDI realmima, pri čemu ih možemo i kombinirati, naime, aplikacije se mogu konfigurirati tako da koriste različite realme.
- Security by obscurity mijenjanjem brojnih defaultnih vrijednosti legitimna je metoda zaštite jer uljeza dovodi u "minsko polje" nepoznanica, veća je vjerojatnost pravovremene detekcije ili će barem ostaviti tragove..., ali bez suradnje i pedantnog dokumentiranja opako ćemo se zamjeriti kolegama.

*

Vijesti: Linux [4]
Kuharice: Za sistemce [5]
Kategorije: Sigurnost [6]
Vote: 0

No votes yet

story_tag: tomcat [7]
Linux [8]
security [9]

zaštita [10]
java [11]

Source URL: https://sysportal.carnet.hr/node/1836

Links
[1] https://sysportal.carnet.hr/node/1834
[2] http://tomcat.apache.org/tomcat-9.0-doc/security-howto.html
[3] https://www.owasp.org/index.php/TLS_Cipher_String_Cheat_Sheet
[4] https://sysportal.carnet.hr/taxonomy/term/11
[5] https://sysportal.carnet.hr/taxonomy/term/22
[6] https://sysportal.carnet.hr/taxonomy/term/30
[7] https://sysportal.carnet.hr/taxonomy/term/285
[8] https://sysportal.carnet.hr/taxonomy/term/119
[9] https://sysportal.carnet.hr/taxonomy/term/129
[10] https://sysportal.carnet.hr/taxonomy/term/204
[11] https://sysportal.carnet.hr/taxonomy/term/288