Linux centralni repozitorij i upravitelj zakrpa

Vugrinec, Matija

Undergraduate thesis / Završni rad

2018

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **Algebra University College / Visoko učilište Algebra**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:225:669291

Rights / Prava: In copyright / Zaštićeno autorskim pravom.

Download date / Datum preuzimanja: **2024-03-28**

Repository / Repozitorij:

Algebra Univerity College - Repository of Algebra Univerity College

VISOKO UČILIŠTE ALGEBRA

ZAVRŠNI RAD

Linux centralni repozitorij i upravitelj zakrpa

Matija Vugrinec

Zagreb, Srpanj 2018.

„Pod punom odgovornošću pismeno potvrđujem da je ovo moj autorski rad čiji niti jedan dio nije nastao kopiranjem ili plagiranjem tuđeg sadržaja. Prilikom izrade rada koristio sam tuđe materijale navedene u popisu literature, ali nisam kopirao niti jedan njihov dio, osim citata za koje sam naveo autora i izvor, te ih jasno označio znakovima navodnika. U slučaju da se u bilo kojem trenutku dokaže suprotno, spreman sam snositi sve posljedice uključivo i poništenje javne isprave stečene dijelom i na temelju ovoga rada".

U Zagrebu, datum.

Ime Prezime

Predgovor

Prvenstveno bih se htio zahvaliti svom mentoru i profesoru Vedranu Dakiću za svu podršku i mentorstvo koje mi je pružio tijekom pisanja završnog rada te zahvaliti na stečenom znanju koje sam dobio kroz njegove kolegije. Htio bih se zahvaliti i stručnom povjerenstvu koje je prepoznalo značaj i korisnost ovog završnog rada. Nikako ne bih htio zaboraviti zahvaliti se i ostalim profesorima koji su me podučavali kroz ove tri godine s posebnim naglaskom na profesora S. Papića koji je svojim predavanjima i vježbama pridonio da uvijek izvučemo još ono malo više znanja.

Izvan akademske zajednice htio bih se zahvaliti dugogodišnjoj curi Maji koja je zajedno sa mnom proživljavala lijepe i one manje lijepe trenutke koji su me pratili kroz akademski i poslovni dio životnog puta. Zahvalio bih se kolegama na poslu koji su bili potpora pri implementaciji Foremana te samom poduzeću Apis-IT koje je omogućilo potrebne resurse. Nikako ne bih htio zaboraviti zahvaliti se roditeljima koji su me podržavali pri odluci da nakon nekoliko godina pauze ipak nastavim visoko školovanje jer smo svi bili svjesni da će se to dugotrajno sigurno isplatiti.

Za kraj zahvalio bih se kolegama s fakulteta zbog kojih sam uvijek davao više od sebe te zbog toga što smo uvijek jedini drugima bili spremni pomoći.

Prilikom uvezivanja rada, Umjesto ove stranice ne zaboravite umetnuti original potvrde o prihvaćanju teme završnog rada kojeg ste preuzeli u studentskoj referadi

Sažetak

U ovom završnom radu instalirao sam i konfigurirao alat Foreman te popratni alat Katello. Foreman i Katello su zajedno poslužili kao centralni Linux repozitorij paketa te kao alati za upravljanje i instaliranje paketima, zakrpama te softverskim nadogradnjama. Foreman je centralni poslužitelj koji je zadužen za upravljanje i instalaciju paketa dok Katello koristim za dohvaćanje paketa iz različitih repozitorija. Značajka ovih dvaju alata je upravljanje s centralnog mjesta stotinama i tisućama poslužitelja bilo da su oni u oblaku (engl. Cloud), hibridnog tipa ili u fizičkoj ili virtualnoj okolini u našem data centru. Ja sam u ovom radu upravljao i nadzirao preko 100-tinjak virtualnih poslužitelja koji se nalaze u prostorijama server sale Apis-IT-a.

Prije same instalacije bitno je bilo postaviti certifikate na poslužitelju te diskove, odnosno particije formatirati sa XFS datotečnim sustavom zbog velikog broja čitanja i pisanja. Kako bih dobio veću fleksibilnost i kako ne bih bio ograničen brojem pisanja i čitanja koristio sam upravljanje logičkim volumenima (engl. Logical Volume Manager, skraćeno LVM).

Nakon same instalacije potrebno je bilo konfigurirati Foreman prema svojoj organizaciji, staviti ime organizacije, odrediti okoline životnih ciklusa, također je bilo potrebno uvesti manifest kako bi Katello mogao pročitati službene Red Hat repozitorije. Virt-who alat sam konfigurirao kako bi mogao dobivati informacije o hipervizorima na kojima se nalaze naši poslužitelji, a sve u svrhu licenciranja istih. Po završetku sinkronizacije repozitorija koje smo uključili po uvođenju manifesta krenuo sam raditi preglede sadržaja (engl. Content View, skraćeno CV) koji su predstavljali same repozitorije. To je bitan korak za kasnije skaliranje sustava. Po izradi svih CV-a, izradio sam zajedničke preglede sadržaja (engl. Composit Content View, skraćeno CCV) koji su se sastojali od različitih CV-a u ovisnosti o potrebama poslužitelja i aplikacija na njima. Nakon izrade CCV-a potrebno je bilo postaviti odgovarajuće aktivacijske ključeve kako bi lako i pomoću jedne poveznice mogli registrirati poslužitelje u odgovarajuće CCV-e te okoline životnih ciklusa.

Po završetku dodavanja svih poslužitelja u Foreman, instalirao sam Remote Execution dodatak koji će mi služiti kao alat za udaljeno i centralno izvođenje naredbi nad poslužiteljima. Poslužitelje sam zbog kasnije jednostavnosti upravljanja podijelio u grupe poslužitelja.

Bez alata poput Foreman-a bilo bi teško ako ne i nemoguće upravljati i nadzirati veći broj Linux poslužitelja.

Ključne riječi: Foreman, Katello, upravljanje, nadzor, centralizirano, Virt-Who, Content-View, Composit Content View

Overview

In my final project I have installed and configured tools Foreman and Katello. Together these two tools have been used as Linux central repositories, they both served as package and patch management. The Foreman is central server that manages installation of packages while Katello is responsible for delivering and connecting appropriate repositories with hosts. The idea of these two tools is to be used to manage hundreds if not thousands of servers from one centralized place whether these servers are located locally in our data center or in the cloud or if we are using hybrid solution. For this project I was managing over one hundred virtual Linux servers that are located in Apis-IT's local data center.

Before the process of installation, I needed to make preparations for companies signed certificates, choose the right file system as any other file system than XFS would not fulfil requirements when it comes to expanding inodes. To get more flexibility when working with disks and partition I've decided to use LVM.

After the installation I needed to tailor Foreman configuration to my organizations needs, it was necessary to configure the name, create life cycle environments, import Red Hat manifest to Katello to stay compliant and to be able to reach repositories. I've installed Virt-Who tool to get connected to VMWare so that Foreman would be able to see hypervisors for licensing. Once I have had imported manifest I started a process of synchronizing repositories and creating content views. Creating content view is an important step for later system scalability. On top of content views, I have created composite content views to accommodate different applications and their needs. To easily register servers to Foreman I have created activation keys which provide a simple link that will place servers in their respective life cycle environments with needed repositories.

After I have registered all servers to Foreman I have installed Remote Execution plugin which enables Foreman to remotely and from centralized location issues commands, installs patches and packages. In order to be more efficient when managing servers, I've created Host collections in which I placed servers based on their attributes and similarities.

Without a tool like Foreman it would be hard and even impossible to manage huge numbers of servers.

Key words: Foreman, Katello, managing, centralized, Virt-Who, Content-View, Composite Content-View

Sadržaj

1. Uvod 1

2. Što je Foreman i koje mogućnosti nudi 2

2.1. Arhitektura sustava 3

2.2. Komponente sustava 5

2.2.1. Katello 5

2.2.2. Candlepin 5

2.2.3. Pulp 5

2.2.4. Hammer CLI 5

2.2.5. REST API 6

2.2.6. Puppet 6

2.3. Organizacije, lokacije i okoline kroz životne cikluse 6

2.3.1. Organizacije 6

2.3.2. Lokacije 6

2.3.3. Okoline kroz životne cikluse 6

2.4. Sadržaj kroz životne cikluse 7

2.5. Koncept poslužiteljskih grupa 8

2.5.1. Organizacije 8

2.5.2. Lokacije 9

2.5.3. Grupe poslužitelja 9

2.5.4. Kolekcije poslužitelja 9

2.6. Struktura poslužiteljskih grupa 9

2.6.1. Ravna struktura 10

2.6.2. Struktura temeljena na životnim ciklusima 10

2.6.3. Struktura orijentirana prema aplikacijama 10
2.6.4. Strukture temeljene prema lokaciji 10
2.7. Koncepti udaljene izrade poslužitelja 11
2.7.1. PXE dizanje sistema 11
2.7.2. Kickstart 12
3. Priprema okruženja za instalaciju Foreman-a 13
3.1. Sistemski zahtjevi 13
3.2. Zahtjevi za prostorom na disku 14
3.3. Konfiguracija prostora na disku prema zahtjevima 16
3.3.1. Izrada logičkih volumena 16
3.3.2. Formatiranje logičkih volumena 17
3.3.3. Montiranje logičkih volumena 18
3.3.4. Povećavanje swap particije 19
3.4. Propuštanje portova na vatrozidu 20
3.5. Konfiguracija SELinux portova 21
3.6. Izrada vlastitog SSL certifikata 23
4. Instalacija Foremana 24
4.1. Dodavanje Red Hat i ostalih repozitorija 24
4.2. Instalacija Foreman-a 25
4.3. Instalacija Virt-Who agenta 27
4.4. Instalacija Hammer CLI dodatka 28
4.5. Instalacija Remote Execution dodatka 29
5. Uređivanje Foremana prema okruženju organizacije 30
5.1. Inicijalna konfiguracija Foremana 30
5.2. Izrada organizacije 32
5.3. Povezivanje s LDAP poslužiteljem 32

5.4. Izrada životnih okolina organizacije 34
5.5. Izrada repozitorija 35
5.5.1. Izrada Red Hat repozitorija 36
5.5.2. Izrada repozitorija trećih strana 37
5.6. Podešavanje rasporeda sinkronizacije repozitorija 40
5.7. Izrada CV-a i kompozitnih CV-ova 40
5.8. Izrada aktivacijskih ključeva 43
5.9. Dodavanje virtualnih poslužitelja u Foreman 44
5.10. Izrada kolekcija poslužitelja 44
6. Udaljeno upravljanje poslužitelja s centralne lokacije 45
6.1. Udaljeno upravljanje paketima 45
6.2. Udaljeno upravljanje zakrpama 46
6.3. Zakazivanje instalacije zakrpi 48
Zaključak 50
Popis kratica 52
Popis slika 53
Popis tablica 55
Popis kôdova 56
Literatura 57

1. Uvod

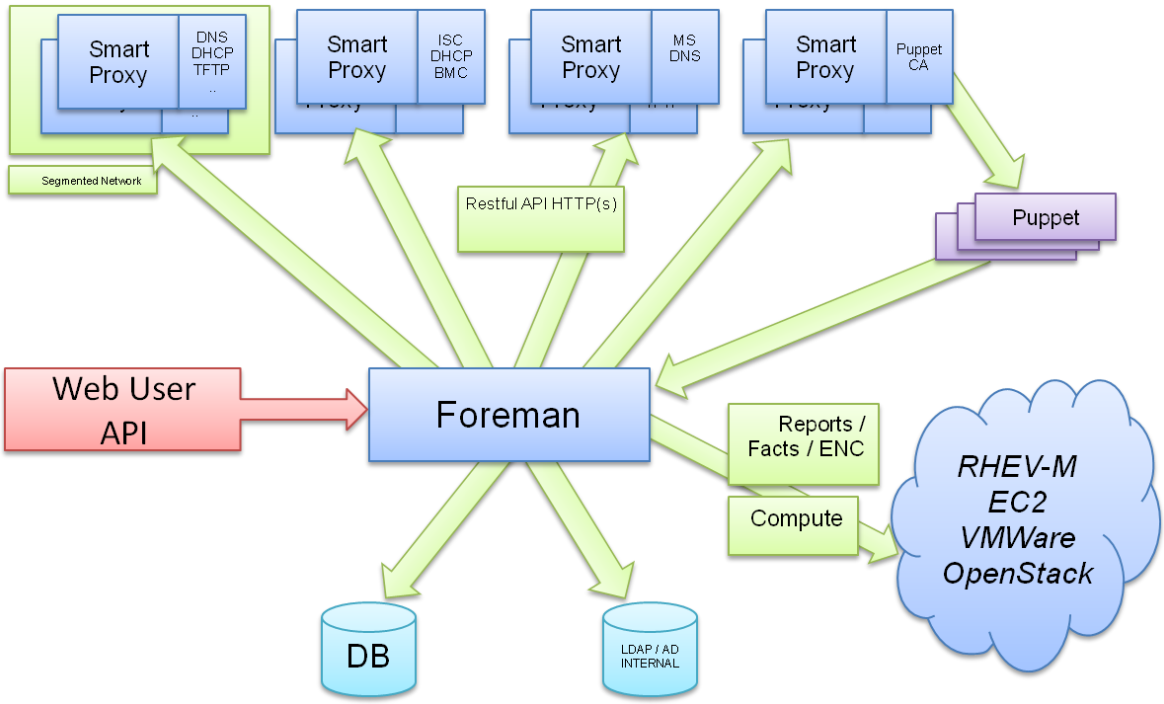
Kako informacijsko tehnološke (engl. Information Technology, skraćeno IT) okoline svakog dana sve više rastu tako postaju sve kompleksnije te ih postaje sve teže nadzirati i upravljati njima. Zaposlenicima ponestaje vremena za administriranje i razvijanje novih rješenja, a organizacije u globalu gledaju gdje mogu smanjiti troškove odnosno biti efikasnije za isti novac.

U takvim scenarijima nastupa Foreman kao alat za pojednostavljivanje upravljanja IT okolinom. Foreman, iako prepun mogućnosti, jednostavan je za korištenje te pomaže organizacijama kod upravljanja cijelim životnim ciklusom svih Linux poslužitelja. Izgrađen je pomoću otvorenih (engl. Open Source) tehnologija koje mu omogućuju automatizaciju i brže upravljanje fizičkim i virtualnim poslužiteljima te poslužiteljima u javnim, privatnim i hibridnim oblacima (engl. Clouds). Pruža granularnu kontrolu nad organizacijom softverskog sadržaja te kako se taj sadržaj propagira kroz različite životne okoline (engl. Lifecycle Environments) koristeći CV-ove. Omogućuje podjelu poslužitelja po grupama za jednostavnije upravljanje ponavljajući zadataka. Pojednostavljuje nadzor sigurnosnih politika uvođenjem Errata gdje se na jednom mjestu mogu vidjeti sve sigurnosne preporuke te bug ispravci (engl. Bug fix). Omogućuje nadzor nad pretplatama, licencama, pokazuje datum isteka istih.

Svrha ovog rada je instalirati i konfigurirati Foreman poslužitelj koji će s jednog centralnog mjesta moći upravljati svim Linux poslužiteljima u organizaciji, udaljeno instalirati te ažurirati različiti softver, dopuštati skrbnicima poslužitelja pristup do službenih repozitorija bez potrebe za Internet vezom, upravljati automatizacijom puštanja zakrpa te sigurnosnih ažuriranja.

2. Što je Foreman i koje mogućnosti nudi

Foreman je projekt otvorenog koda (engl. Open source) koji pomaže sistem administratorima upravljati poslužiteljima kroz sve životne cikluse, od nadgledanja (engl. provisioning), konfiguracije, orkestriranja do nadzora i upravljanja pretplatama na licence. Koristeći Puppet, Chef, Salt ili Ansible i Forman-ovu *Smart-proxy* arhitekturu (Slika 2.1), lako je automatizirati ponavljajuće zadatke, brzo isporučiti aplikaciju (engl. deploy) i proaktivno upravljati promjenama bilo da su virtualni ili fizički poslužitelji u vlastitom podatkovnom centru ili u oblaku. Foreman pruža opsežne i interaktivne servise (engl. facilities) poput web sučelja koji se pokreću na Apache web poslužitelju, komandno linijskog sučelja (engl. Command-line Interface, skraćeno CLI) zvanog Hammer CLI te *RESTful* aplikacijskog programskog sučelja (engl. Application Program Interface, skraćeno API) koji omogućuju veću razinu poslovne logike.



Slika 2.1 Foreman Smart-Proxy arhitektura (izvor www.theforeman.org)

Ukratko, Foreman omogućuje:

- Otkrivanje, nadgledanje i nadogradnje cjelokupne *bare-metal* infrastrukture
- Kreiranje i upravljanje instancama preko privatnih i javnih oblaka

- Grupiranje poslužitelja i njihovo grupno (engl. bulk) upravljanje bez obzira na lokaciju
- Pregled svih promjena na poslužiteljima zbog revizije ili rješavanja problema (engl. troubleshooting)
- Proširivanje mogućnosti dodavanjem dodataka (engl. plugins)
- Automatizirano kreiranje slika (engl. image) za različite sustave kako bi se optimiziralo dodavanje novih poslužitelja (eng. deploy)

2.1. Arhitektura sustava

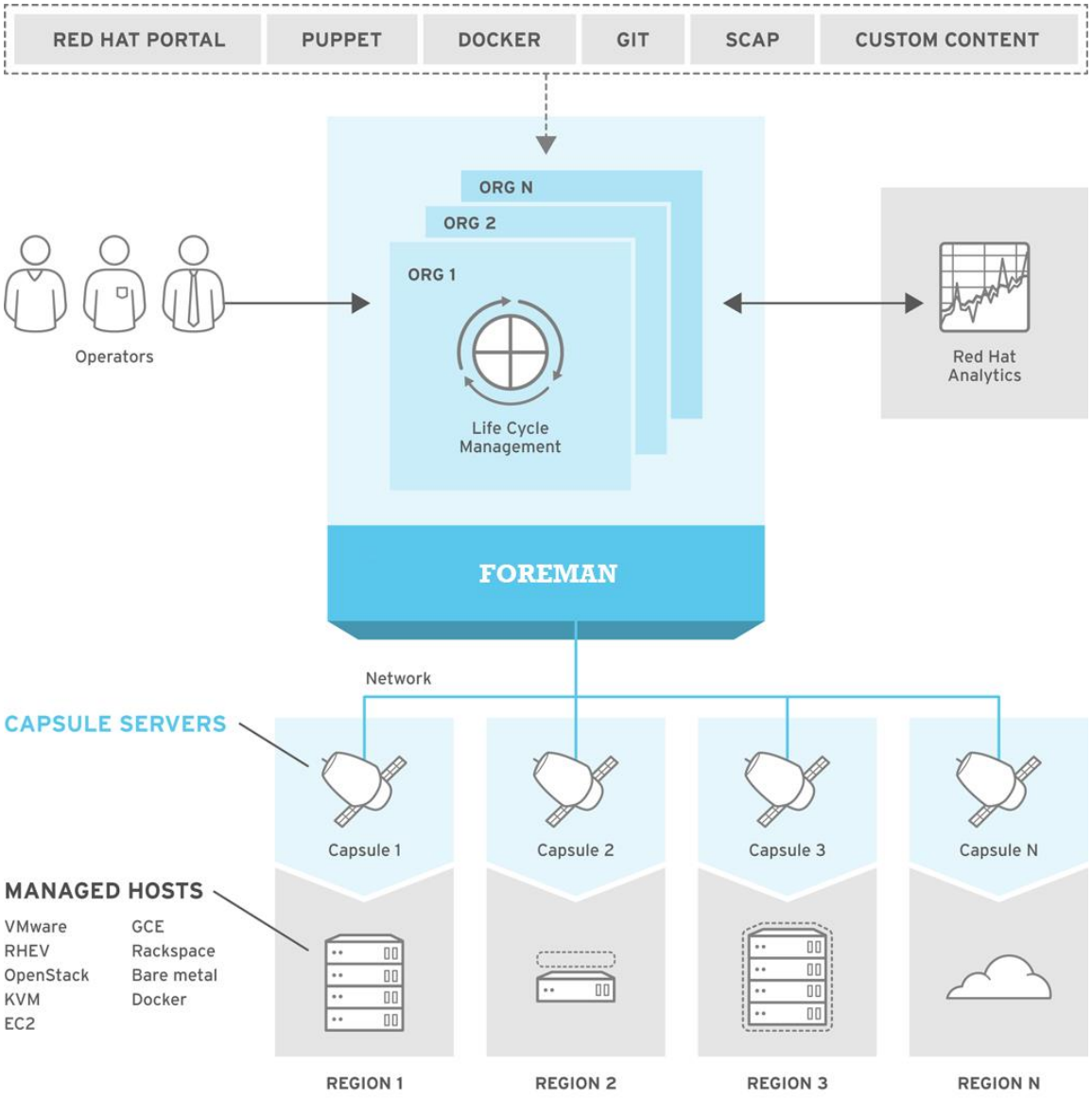
Foreman arhitektura podržava četiri nivoa kroz koje teku podaci (Slika 2.2):

- Vanjski izvor podataka
  - Foreman može konzumirati različite vrste podataka iz različitih izvora. Ako imamo Red Hat licencu tada Red Hat može biti izvor sigurnih softver paketa, Errata, Puppet modula i slika za kontejnere. Drugi izvori mogu biti Git repozitorij, Docker Hub, Puppet Forge, te također organizacija može sama izraditi svoj lokalni repozitorij.
- Foreman poslužitelj
  - Foreman omogućuje planiranje i upravljanje kroz različite životne cikluse kapsula poslužitelja i upravljanih poslužitelja koristeći web grafičko sučelje (engl. Graphical User Interface, skraćeno GUI), CLI ili API.
  - Foreman organizira upravljanje životnim ciklusom koristeći Organizacije kao glavnu jedinicu podjele. Organizacije izoliraju sadržaj za grupu poslužitelja sa specifičnim zahtjevima i administracijskim zadacima.
  - Također omogućuje finu granulaciju autentikacijskih prava koje omogućuju ili zabranjuju administratorima pristup pojedinim dijelovima infrastrukture.
- Poslužitelj kapsula (engl. Capsule server)
  - Preslikava sadržaj glavnog Foreman servera na različite geografske lokacije kako bi omogućili poslužiteljima na tim lokacijama da povuku sadržaj s tog poslužitelja umjesto da dodatno opterećuju mrežu i traže sadržaj s centralne lokacije.
  - Koristeći CCV detaljnije se može granulirati sadržaj koji kapsula poslužitelj može ponuditi.

- Omogućuje komunikaciju između upravljanih poslužitelja i Foremana tako da sve aplikacije i svi servisi mogu doći do Foremana kroz jednu IP adresu što pojednostavljuje administraciju vatrozida.

▪ Upravljani poslužitelji

- To su poslužitelji koji mogu preuzimati sadržaj od kapsula poslužitelja
- Mogu biti fizički ili virtualni poslužitelji na raznim platformama poput KVM, VMware vSphere, OpenStack i drugih.
- Mogu biti direktno povezani s Foreman poslužiteljem



Slika 2.2 Foreman arhitektura sustava (izvor www.redhat.com)

2.2. Komponente sustava

Foreman sustav se sastoji od nekoliko projekata otvorenog koda koji su međusobno integrirani, testirani te rade kao jedan sustav.

Ovo su ti projekti:

2.2.1. Katello

Katello je Foreman dodatak koji služi za upravljanje pretplatama i repozitorijima. Omogućuje preuzimanje sadržaja iz različitih repozitorija, stvaranje, organiziranje i upravljanje lokalnim repozitorijem. Sinkronizira udaljene repozitorije, učitava (engl. upload) sadržaj kako bi se stvorile biblioteke (engl. library) sadržaja iz kojih se kasnije mogu stvarati CV-ovi.

Omogućuje filtriranje paketa, errata i puppet modula iz lokalnog sadržaja kako bi se stvorili CV-ovi koji mogu biti promovirani kroz različite životne cikluse razvoja. Primjer toga bi bilo promoviranje CV iz razvojne okoline u testnu, pa kasnije u produkcijsku okolinu.

Koristi Katello Agent koji se instalira na željene poslužitelje kako bi dobili više informacija o erratama.

2.2.2. Candlepin

Candlepin je servis unutar Katello-a koji se brine za upravljanje pretplatama na siguran način.

2.2.3. Pulp

Pulp je servis unutar Katello-a koji upravlja sadržajem i repozitorijima. Osigurava da se diskovni prostor koristi efikasno na način da ne duplicira pakete čak i kada su ti paketi zatraženi iz različitih organizacija.

2.2.4. Hammer CLI

Hammer CLI je alat koji omogućuje sve pa čak i više funkcija za upravljanje Foremanom od web korisničkog sučelja. Potrebno ga je naknadno instalirati jer dolazi kao dodatak, ali je neizmjerno koristan kod zadataka koji se mogu skriptirati

2.2.5. REST API

Foreman dolazi sa RESTful API servisom koji omogućuje administratorima i razvojnim timovima da razvijaju vlastite skripte i *third-party* aplikacije.

2.2.6. Puppet

Foreman se oslanja na Puppet za konfiguraciju poslužitelja, za automatizaciju ponavljajućih zadataka, za isporuku aplikacija i upravljanje promjenama na poslužiteljima. Puppet također skuplja preko svojeg agenta različite podatke koji se kasnije mogu vidjeti u Foremanu u obliku različitih tablica i statistika.

2.3. Organizacije, lokacije i okoline kroz životne cikluse

Foreman poslužitelj konsolidira pristup upravljanju organizacija i lokacija. Sistem administratori mogu stvoriti više organizacija i više lokacija unutar Foreman poslužitelja te stvarati različite kontekste u ovisnosti o hijerarhiji ili nekom drugom pristupu. Na različite lokacije se postavljaju kapsula poslužitelji koji sinkroniziraju sadržaj te upravljaju konfiguracijama na tim lokacijama.

2.3.1. Organizacije

Organizacije razdvajaju resurse po logičkim grupama koje se mogu temeljiti na vlasništvu, svrsi, prema sadržajima, sigurnosnom nivou ili nekom drugom pristupu.

2.3.2. Lokacije

Lokacije razdvajaju organizacije u logičke grupe u ovisnosti o geografskoj lokaciji. Svaka sadrži jedan kapsula poslužitelj kojim upravlja Foreman.

2.3.3. Okoline kroz životne cikluse

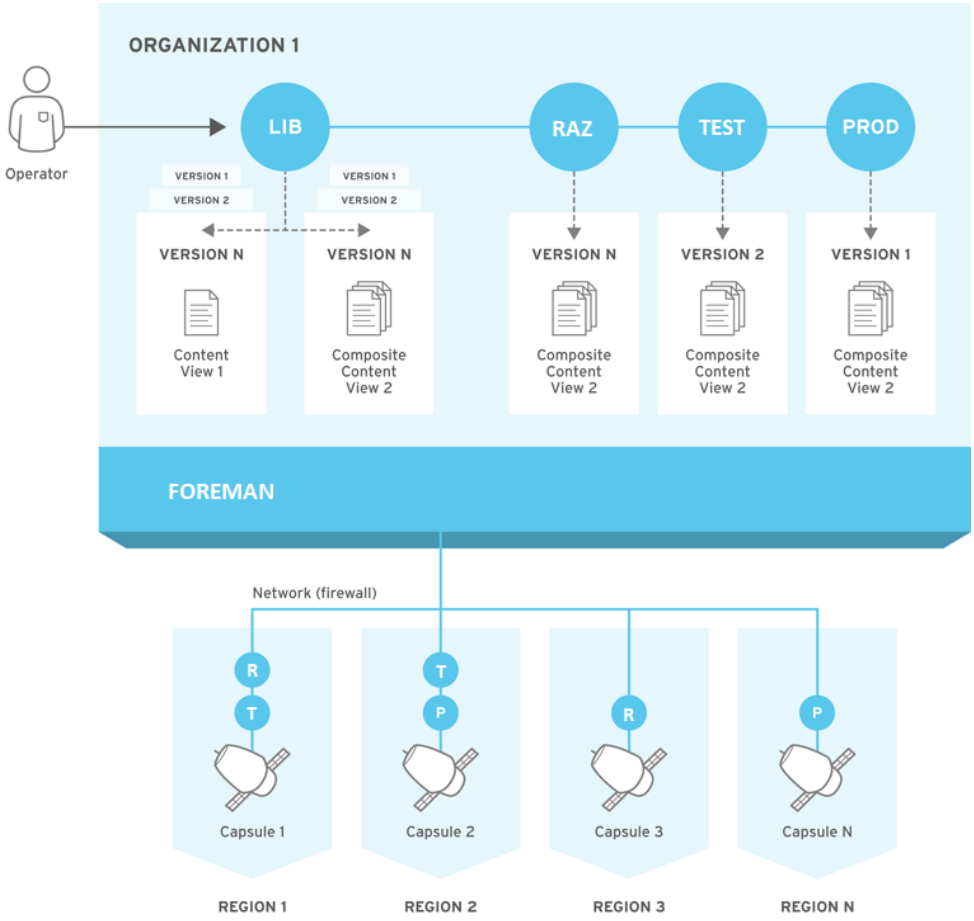
Aplikacije su podijeljene prema okolinama u ovisnosti o životnom ciklusu u kojem se izvode. Okoline kroz životne cikluse su zajedno povezane te tvore put okolina (engl. environment path). Moguće je promovirati sadržaj koji se trenutno nalazi u primjerice razvojnoj okolini u testnu okolinu po potrebi.

2.4. Sadržaj kroz životne cikluse

Nakon što smo registrirali Foreman ili kapsula poslužitelj na neki od repozitorija, sav sadržaj koji možemo preuzeti predefinirano se smješta u *Library* pregled sadržaja. Takav smještaj nam nužno ne mora odgovarati i najčešće ćemo htjeti napraviti vlastite CV-ove.

CV-ovi su podskupovi sadržaja koje smo izdvojili filtriranjem iz *Library-a* i kao takve možemo ih objaviti (engl. Publish) i promovirati (engl. Promote) u različite okoline. Primjerice, trenutno testiramo neku aplikaciju u razvojnoj i testnoj okolini te smo završili s testiranjem u razvoju, sada tu verziju CV-a možemo promovirati u testnu okolinu na daljnje testiranje.

CV-ovi se mogu kombinirati u CCV, ovaj pristup je koristan kada imamo repozitorij za operacijski sustav te repozitorij za aplikaciju koja se izvodi na tom poslužitelju. Jedna od prednosti je ta da kada dođe nova verzija paketa u jednom od ta dva repozitorija, samo CV u kojem se nalazi taj repozitorij je potrebno ponovo objavit. Nakon toga u CCV samo stavimo taj ponovno objavljeni CV, drugi ne moramo dirati.



Slika 2.3 Pregled sadržaja kroz životne cikluse (izvor www.redhat.com)

2.5. Koncept poslužiteljskih grupa

Uz fizičke topologije koje nudi Foreman, nudi i nekoliko logičkih jedinica za grupiranje poslužitelja. Poslužitelji koji su članovi tih grupa mogu nasljeđivati grupne konfiguracije. Glavne logičke grupe su podijeljene kao organizacije, lokacije, grupe poslužitelja (engl. Host groups) i kolekcije poslužitelja (engl. Host collections). Lokacije i grupe poslužitelja mogu biti ugniježđene dok su organizacije i kolekcije poslužitelja

2.5.1. Organizacije

Najveći nivo logičkih grupa poslužitelja. Organizacije pružaju vrlo distinktivnu separaciju sadržaja i konfiguracija, svaka organizacija zahtjeva svoje pretplatničke manifeste ako imamo licence i smatra se kao posebna virtualna instanca Foreman poslužitelja.

2.5.2. Lokacije

Grupe poslužitelja trebaju odgovarati stvarnoj fizičkoj lokaciji na kojoj se nalaze. Lokacije se koriste kako bi se odredilo kojoj mrežnoj infrastrukturi poslužitelj pripada kako bi se spriječila pogrešna konfiguracija poslužitelja. Primjerice, mrežne informacije poput podmreže (engl. subnet), domene se ne mogu dodijeliti direktno kapsula poslužitelju već lokaciji na kojoj se taj poslužitelj nalazi.

2.5.3. Grupe poslužitelja

Grupe poslužitelja su glavni nosioci definicije poslužitelja uključujući dodijeljenu Puppet klasu, CV ili operacijski sustav (engl. Operating System, skraćeno OS). Grupe poslužitelja se koriste kod stvaranje kompletno novog poslužitelja kojeg Foreman nadzire od samog početka. U skladu s time preporuča se napraviti kompletnu konfiguraciju poslužitelja ovdje umjesto da se na svakom poslužitelju radi posebno. Ako se koriste grupe poslužitelja tada je dodavanje novog poslužitelja jednostavno jer ga moramo staviti samo u grupu prema kojoj će dalje naslijediti prava i konfiguracije.

2.5.4. Kolekcije poslužitelja

Svi poslužitelji koji su registrirani u Foreman zbog dostupnosti sadržaja se zovu *Content host*. Ti poslužitelji se stavljaju u kolekcije poslužitelja kako bi kasnije bilo jednostavnije nad njima izvoditi grupne akcije instaliranja paketa i errata.

2.6. Struktura poslužiteljskih grupa

Mogućnost da poslužiteljske grupe mogu biti ugniježdene i tako nasljeđivati prava i konfiguracije dopušta nam da dizajniramo strukture poslužitelja kako to najbolje odgovara našoj organizaciji. Dobro isplanirani dizajn strukture poslužitelja može uvelike pomoći kod kasnijeg održavanja poslužitelja. Četiri su glavne strukture poslužitelja, ravna (engl. Flat), struktura temeljena na životnim ciklusima, struktura orijentirana prema aplikacijama i prema lokacijama na kojima se nalaze poslužitelji (Slika 2.4).

2.6.1. Ravna struktura

Prednost ovakve strukture je njena jednostavnost jer ne postoji nasljeđivanje. U organizaciji s malim brojem poslužitelja ovo bi bila najbolja opcija, ali treba biti dodatno oprezan da ne dođe do dupliciranja postavki.

2.6.2. Struktura temeljena na životnim ciklusima

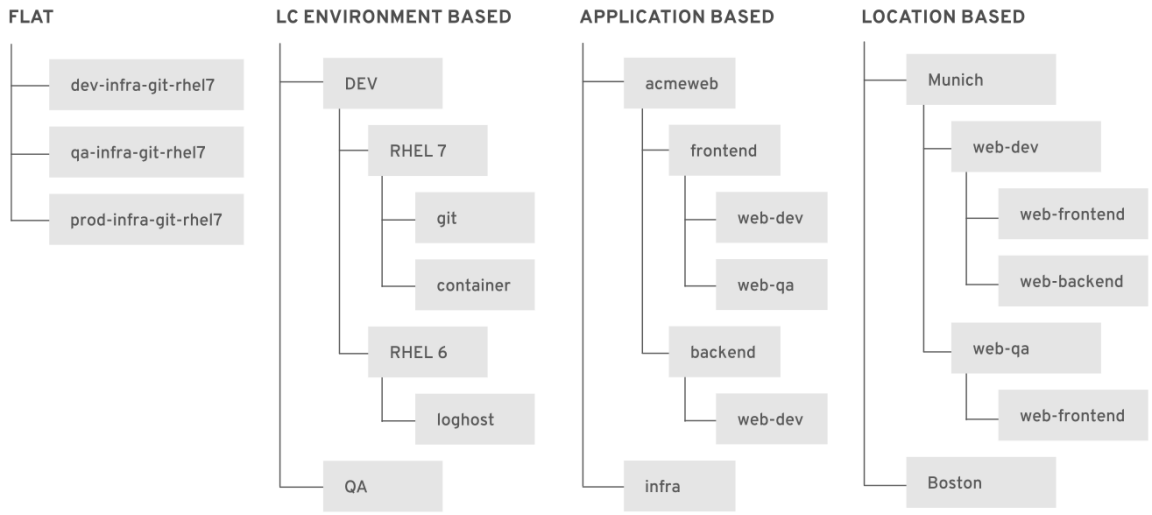
Radi se o hijerarhijskom tipu strukture, prvi nivo sadrži parametre specifične za okolinu životnog ciklus. Nivo ispod je rezerviran za parametre OS-a dok je treći i posljednji nivo zadužen za parametre specifične aplikacije. Takva struktura je korisna kod scenarija kada su odgovornosti podijeljene kroz okoline životnog ciklusa.

2.6.3. Struktura orijentirana prema aplikacijama

Hijerarhijska struktura temeljena je na ulogama poslužitelja u određenim aplikacijama. Primjerice, uloga koja definira mrežne postavke za *back-end* i *front-end* poslužitelje. Ovdje su karakteristike poslužitelja podijeljene te su kao takve idealne za upravljanje pomoću Puppeta.

2.6.4. Strukture temeljene prema lokaciji

U ovoj hijerarhijskoj strukturi lokacije su usklađene sa strukturom poslužiteljskih grupa. U scenariju gdje lokacija odnosno kapsula poslužitelj određuje mnoge atribute, ovakav pristup je najbolji. S druge strane ovakva struktura otežava dijeljenje parametara s drugim lokacijama koje imaju malo kompleksnija okruženja.



Slika 2.4 Struktura poslužiteljskih grupa (izvor www.redhat.com)

2.7. Koncepti udaljene izrade poslužitelja

Još jedna važna mogućnost Foremana je bez nadzorno (engl. unattended) izrađivanje (engl. provisioning) poslužitelja. Kako bi nešto takvo bilo moguće, Foreman koristi infrastrukturu domenskog sustava imena (engl. Domain Name Service, skraćeno DNS) i protokol za dinamičko konfiguriranje računala (engl. Dynamic Host Configuration Protocol, skraćeno DHCP), okruženje izvođenja prije dizanja sistema (engl. Preboot Execution Environment, skraćeno PXE), jednostavan protokol za prijenos datoteka (engl. Trivial File Transfer Protocol, skraćeno TFTP) i Kickstart.

2.7.1. PXE dizanje sistema

PXE omogućava dizanje sistema preko mreže, umjesto korištenja lokalnih tvrdih diskova. Koristi DHCP kako bi poslužitelj preko mreže dobio sve potrebne mrežne postavke, koristi DNS kako bi pronašao TFTP poslužitelj s kojeg preuzima sliku za podizanje sustava. Za uspješno PXE podizanje potrebno je prema DHCP-u i TFTP-u imati otvorene UDP portove 67, 68 i 69 te TCP port 80 kako bi se preuzeli Kickstart predlošci sa kapsula poslužitelja.

2.7.2. Kickstart

Kickstart instalacija nudi mogućnost automatizacije djelomičnog ili cijelog instalacijskog procesa. Kickstart konfiguracijska datoteka sadrži odgovore na sva pitanja koja pita instalacijski program, poput vremenske zone, koji sistem želimo instalirati, kako trebaju biti particirani diskovi ili koje pakete želimo instalirati. Ako koristimo takvu konfiguracijsku datoteku pri instalaciji, ta instalacija može biti kompletno bez nadzora, odnosno nije potrebna nikakva ljudska intervencija. Takav način instalacije je koristan kada se odjednom treba instalirati veći broj poslužitelja.

3. Priprema okruženja za instalaciju Foreman-a

Zbog toga što je Foreman vrlo kompleksan proizvod koji se sastoji od mnogih servisa, vrlo je bitno proučiti dokumentaciju te pripremiti okruženje za instalaciju kako bismo izbjegli moguće probleme kod budućeg korištenja. Nažalost, dokumentacija je vrlo štura pa se događa da se tijekom korištenja Foremana pojave neke greške koje su se lako mogle izbjeći dobrom pripremom. Priprema okruženja dakako ovisi o broju poslužitelja koje planiramo nadzirati, broju repozitorija te o samom hardveru kojeg koristimo.

3.1. Sistemski zahtjevi

Slijedeća tablica prikazuje minimalne sistemske zahtjeve u usporedbi s konfiguracijom poslužitelja kojeg sam koristio.

Zahtjevi	Minimalni sistemski zahtjevi	Foreman poslužitelj
Arhitektura	64-bit	64-bit
Verzija operacijskog sustava	CentOS 7 ili RHEL 7	RHEL 7.4
Broj procesorskih jezgra	2	4
Radna memorija	12GB	16GB
Swap memorija	4GB	4GB

Tablica 3.1 Sistemski zahtjevi Foreman poslužitelja

Kada sam testirao Foreman u instalacijskoj okolini, koristio sam minimalne sistemske zahtjeve što je rezultiralo sporijim radom poslužitelja, no također sam primijetio da su se i češće pojavljivale greške kada je poslužitelj bio pod punim opterećenjem. Te greške su se manifestirale prekidima promocija CV-a ili pak predugim trajanjima promocija što je uzrokovalo gubitak vremena te ručno brisanje zadataka u Postgres bazi. Kako je Foreman jedan od ključnih poslužitelja u našoj produkcijskoj okolini odlučio sam se povećati broj procesorskih jezgri te dodati dodatnih 4GB radne memorije kako bi se izbjegle prije objašnjene greške.

3.2. Zahtjevi za prostorom na disku

Zahtjeve za prostorom na disku sam izdvojio kao posebnu cjelinu jer na njih treba obratiti dodatnu pažnju te zahtijevaju dodatnu konfiguraciju. Kod računanja potrebnog prostora na disku moramo uzeti u obzir broj repozitorija koji ćemo sinkronizirati. Kako u početku ne možemo znati točan broj zbog toga što će taj broj s vremenom i potrebama rasti, trebamo se na to pripremiti tako da možemo vrlo brzo i lako povećavati prostor. S povećavanjem prostora otvara se pitanje koji datotečni sustav koristiti kako bi mogli podržati veliki broj pisanja i čitanja.

Svaki puta kada sinkroniziramo novi repozitorij, svi paketi unutar tog repozitorija se preuzmu na Foreman poslužitelj, ako imamo više repozitorija s istim paketima, ti paketi se ne dupliciraju. Paketi koji se preuzmu spremaju se u `/var/lib/mongodb` i `/var/lib/pulp` direktorije te trebamo osigurati da ti direktoriji ne ostanu bez slobodnog prostora. Prilikom preuzimanja paketa `/var/cache/pulp` direktorij privremeno pohranjuje sadržaj dok se taj sadržaj ne sinkronizira. Za sadržaj u paketima (engl. Red-hat Package Manager, skraćeno RPM), sprema maksimalno 5 RPM-ova koje nakon prelaska ograničenja prebacuje u `/var/lib/pulp`. ISO (engl. International Organization for Standardization) format sadržaja pohranjuje u `/var/cache/pulp` dok se sve ne preuzme pa tek onda prebacuje taj sadržaj u `/var/lib/pulp`. Slijedeći bitan direktorij je `/var/lib/qpidd` u koji se pohranjuje sadržaj za svakog klijenta, u praksi za svakog klijenta potrebno 2 megabajta (engl. Megabyte, skraćeno MB) prostora.

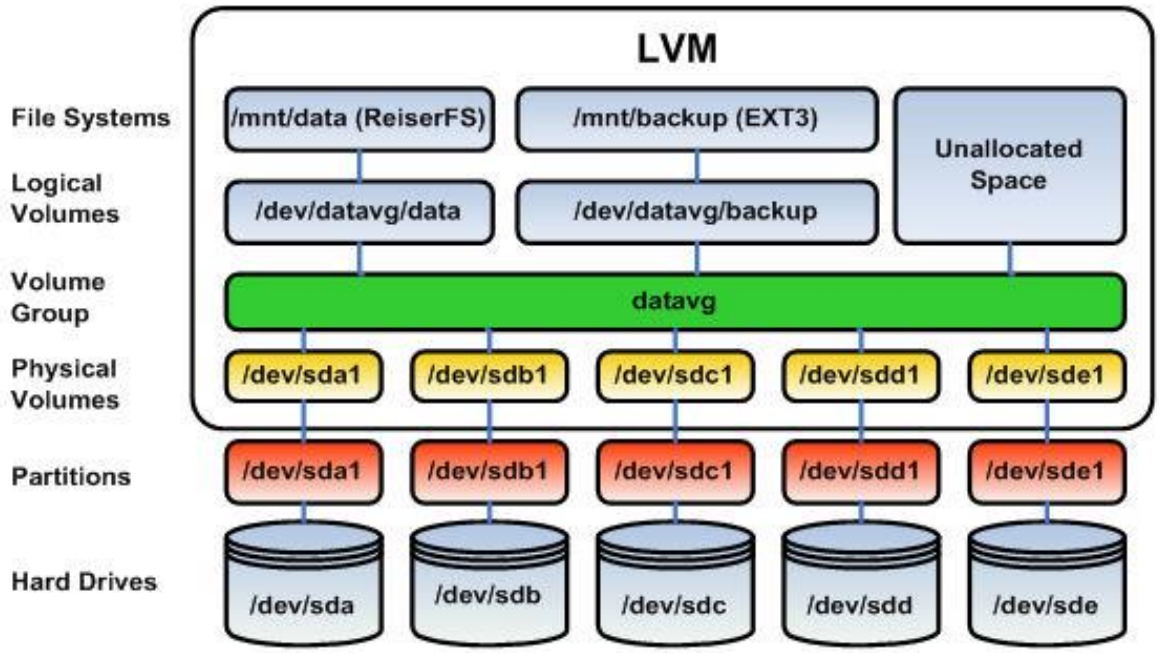
Svima gore navedenim direktorijima treba dati onoliko prostora koliko to naša okolina zahtjeva. U slijedećoj tablici naveo sam koliko prostora je dodijeljeno Foreman poslužitelju te napomene kojih bi se trebalo držati prilikom planiranja.

Direktorij	Količina prostora	Napomene
/var/lib/pulp	90GB	Planirati rast prostora sa širenjem infrastrukture. Ne podržava simboličke linkove.
/var/cache/pulp	10GB	Privremeno pohranjuje sadržaj.
/var/lib/mongodb	30GB	Planirati rast prostora sa širenjem infrastrukture. Ne podržava simboličke linkove. Ne preporuča se korištenje mrežnog datotečnog sustava (engl. Network File System, skraćeno NFS).
/var/spool/squid	12GB	Služi kao privremena lokacija kod preuzimanja repozitorija prije sinkronizacije. Ima tendenciju rasta ako se preuzima više repozitorija odjednom.
/var/lib/pgsql	10GB	Planirati rast prostora sa širenjem infrastrukture. Služi za PostgreSQL bazu. Ne preporuča se korištenje NFS-a.

Tablica 3.2 Količina prostora podijeljena po ključnim direktorijima

3.3. Konfiguracija prostora na disku prema zahtjevima

Zahtjevi Foreman poslužitelja su vrlo dinamični te imamo konstantno povećanje broja poslužitelja u našem sistemskom centru. Prema toma treba izabrati ispravan način kojim ćemo lako povećavati prostor prema potrebi. Kao logično rješenje nameće se korištenje LVM-a gdje vrlo lako i jednostavno možemo dodavati prostor i isto tako ga oduzimati, premda tada treba uzeti u obzir koji datotečni sustav koristimo. LVM je alat za upravljanje logičkim volumenima koji uključuje alociranje diskova, njihovo zrcaljenje (engl. mirroring), *striping* te povećavanje odnosno smanjivanje logičkih volumena. LVM koristi set različitih diskova koje podijelimo u fizičke volumene, nakon toga izradimo grupe volumena u koje stavimo količinu prostora koji želimo imati te kasnije možemo iz tih grupa volumena kreirati logičke volumene.



Slika 3.1 Prikaz LVM sheme (izvor linux.macrofinances.com)

3.3.1. Izrada logičkih volumena

Prije nego krenemo u izradu logičkih volumena potrebno je naredbom `fdisk -l` dobiti popis diskova koje imamo na raspolaganju. Svi diskovi koji su prikazani našem serveru su vidljivi u obliku /dev/sda, /dev/sdb i tako dalje u ovisnosti koliko ih imamo. Kako bi particionirali disk potrebno je koristiti ovu naredbu `fdisk /dev/sdb`, pritisnuti slovo **n** kako započeli izradu nove particije. Na upit interaktivnog izbornika potrebno je odgovoriti

slovom **p** koje označuje da ćemo koristiti tu particiju kao primarnu za taj disk, nakon toga pritisnuti tipku enter dva puta kako bi potvrdili broj particije te da će particija zauzeti sav prostor na disku. Nakon tih koraka moramo particiju označiti tipom koji je pogodan za LVM, a to radimo tako da unesemo slovo **t** te na upit poslužitelja unesemo **8e**. Kako bi provjerili učinjeno do sada, unesemo slovo **p** i ako smo zadovoljni učinjenim unesemo slovo **w** koje će promjene zapisati. Isti postupak ponovimo za sve diskove koje želimo koristiti te ponovo naredbom `fdisk -l` provjerimo novo stanje diskova i novonastalih particija.

Nakon izrade particija spremni smo za izradu fizičkih volumena koje radimo naredbom `pvcreate /dev/sdb1`, obavezno stavimo punu putanju particije. Kada smo izradili fizički volumen tada možemo izraditi grupu volumena. Naredba je dosta slična no ovdje trebamo zadati ime volumen grupe na koju ćemo se kasnije referencirati kod izrade logičkih volumena. Naredbom `vgcreate foremanvg /dev/sdb1` stvaramo grupu volumena. U slučaju kada nam ponestane prostora na grupi volumena, jednostavno možemo predstavit novi disk našem poslužitelju te proširiti grupu volumena naredbom `vgextend foremanvg /dev/sdc1` gdje je /dev/sdc1 novi fizički volumen.

Kada smo izradili grupe volumena možemo krenuti u izradu logičkih volumena. Logički volumeni se rade naredbom `lvcreate` te se dodaju po potrebi zastavice kojima govorimo poslužitelju detalje. Standardna zastavica je -n kojom zadajemo ime logičkom volumenu te -L kojim specificiramo količinu prostora koji dodajemo u taj logički volumen. Ovo su logički volumeni i količina prostora koje sam ja izradio na Foreman poslužitelju:

```
lvcreate -n squidlv -L +5G foremanvg
lvcreate -n pulplv -L +90G foremanvg
lvcreate -n mongolv -L +30G foremanvg
lvcreate -n pulpcachelv -L +10G foremanvg
lvcreate -n pgsqllv -L +10G foremanvg
```

Kod 3.1 Izrada logičkih volumena

3.3.2. Formatiranje logičkih volumena

Foreman prilikom preuzimanja, sinkronizacije te objavljivanja novih CV-ova i repozitorija ima velik broj pisanja po disku. Također koristi velik broj simboličkih linkova te zbog tih likova i velikog broja pisanja troši velik broj *inode*-ova. *Inode* je podatkovna struktura u Linux datotečnim sustavima koja opisuje objekte datotečnih sustava poput podatka ili

direktorija. Svaki *inode* sprema atribute i lokacije blokova gdje se neki podatak nalazi. Atributi podatka mogu sadržavati meta podatke poput vremena zadnje promjene, same promjene ili pristupa podatku. Vrlo je bitno da ne ostanemo bez slobodnog prostora na logičkim volumenima, ali isto tako da ne ostanemo bez slobodnih *inode*-ova. Kako se to ne bi dogodilo, odnosno kako bi mogli utjecati na to, logičke volumene moramo formatirati kao XFS datotečni sustav. XFS nam za razliku od EXT4 i ranijih datotečnih sustava omogućuje povećanje broja *inode*-ova. Bitna stvar za napomenuti kod formatiranja logičkih volumena XFS datotečnim sustavom je da se ti logički volumeni mogu proširivati dodatnim prostorom, no ako smo dodijelili previše prostora nije ga više moguće vratiti u grupu volumena jer XFS ne podržava smanjivanje prostora i zasigurno ćemo pokvariti taj logički volumen i nepovratno izgubiti datoteke na njemu što bi u slučaju mongolv-a nepovratno uništilo Foreman poslužitelj. Za formatiranje logičkih volumena koristio sam slijedeće naredbe:

```
mkfs.xfs /dev/foremanvg/squidlv
mkfs.xfs /dev/foremanvg/pulplv
mkfs.xfs /dev/foremanvg/mongolv
mkfs.xfs /dev/foremanvg/pulpcachelv
mkfs.xfs /dev/foremanvg/pgsqllv
```

Kod 3.2 Formatiranje logičkih volumena

3.3.3. Montiranje logičkih volumena

Nakon što smo izradili logičke volumene te ih formatirali traženim datotečnim sustavom potrebno ih je montirat (engl. mount) u datotečni sustav kako bi ih Foreman nakon instalacije mogao koristiti. Zbog toga što unaprijed znamo koje direktorije Foreman koristi, možemo ih prije instalacije izraditi pomoću `mkdir` naredbe.

```
mkdir /var/spool/squid
mkdir /var/lib/pulp
mkdir /var/lib/mongodb
mkdir /var/cache/pulp
mkdir /var/lib/pgsql
```

Kod 3.3 Izrada direktorija za montiranje

Nakon same izrade direktorija koje ćemo montirati (engl. mountpoint) potrebno je u konfiguracijsku datoteku **fstab** staviti upute operacijskom sustavu navodeći mu koje datoteke treba montirati po ponovnom paljenju sustava.

```
vi /etc/fstab
/dev/mapper/foremanvg-pulplv /var/lib/pulp       xfs
defaults     1 2
/dev/mapper/foremanvg-squidlv /var/spool/squid   xfs
defaults      1 2
/dev/mapper/foremanvg-mongolv /var/lib/mongodb   xfs
defaults      1 2
/dev/mapper/foremanvg-pulpcachelv /var/cache/pulp xfs
defaults      1 2
/dev/mapper/foremanvg-pgsqllv /var/lib/pgsql    xfs
defaults      1 2
```

Kod 3.4 Podešavanje konfiguracije u /etc/fstab datoteci

Na kraju je potrebno montirati direktorije `mount` naredbom.

```
mount /var/spool/squid/
mount /var/lib/pulp
mount /var/lib/mongodb/
mount /var/cache/pulp/
mount /var/lib/pgsql/
```

Kod 3.5 Mount direktorija

3.3.4. Povećavanje swap particije

Kod minimalnih zahtjeva navodi se da je potrebno imati barem 4GB prostora za swap memoriju. *Swap* memorija se koristi kada se zapuni kapacitet radne memorije na poslužitelju, tada se neaktivne stranice u memoriji prebacuju u *swap* prostor koji smo alocirali. *Swap* memorija se prije koristila znatno češće, no i sada se koristi kada nemamo dovoljno radne memorije na raspolaganju. Prostor koji smo alocirali kao swap memoriju zapravo koristimo prostor diskovnog kapaciteta, zbog toga je swap memorija sporija od radne memorije. Kako ja sam stavio 16GB radne memorije, odnosno 4GB više od minimalne, ni kod najvećih opterećenja sustava nisam primijetio da je da je swap memorija bila znatnije korištena pa nisam ni imao potrebe povećavati taj prostor. Kako je inicijalno bilo na poslužitelju konfigurirano samo 2GB swap memorije, morao sam dodijeliti još 2GB što samo učinio ovim naredbama;

```
swapoff -v /dev/rootvg/swap
lvresize /dev/rootvg/swap -L +2G
mkswap /dev/rootvg/swap
swapon -v /dev/mapper/rootvg-swap
```

Kod 3.6 Povećavanje swap particije

3.4. Propuštanje portova na vatrozidu

Određeni mrežni portovi moraju biti otvoreni i slobodni na operacijskom sustavu poslužitelja kako bi komponente Foreman poslužitelja, kapsula poslužitelja i klijenata mogle međusobno komunicirati. Ako ti portovi nisu otvoreni i slobodni prije instalacije, tada će instalacija Foremana biti neuspješna. U slijedećoj tablici naveo sam portove koje sam otvorio na vatrozidu operacijskog sustava.

Port	Protokol	Servis	Potreba
80	TCP	HTTP	Služi za preuzimanje Katello certifikata, predložaka.
443	TCP	HTTPS	Spajanje na Red Hat pretplatničku stranicu kako bi preuzeo manifest. Pristup Foreman web stranici. Spajanje na Katello agenta. Spajanje na Katello, Foreman, Foreman API, Puppet.

5647	TCP	AMQP	Komunikacija Katello agenta sa Qpid preklopnikom.
8140	TCP	HTTPS	Komunikacija Puppet agenta s Puppet poslužiteljem.
22	TCP	SSH	Komunikacija koja omogućuje udaljeno upravljanje poslužiteljima.

Tablica 3.3 Propušteni portovi na vatrozidu

Naredbe koje sam koristio za propuštanje portova kroz vatrozid su slijedeće:

```
firewall-cmd --permanent \
--add-port="22/tcp" \
--add-port="80/tcp" \
--add-port="443/tcp" \
--add-port="5647/tcp" \
--add-port="8140/tcp"
```

Kod 3.7 Propuštanje portova kroz vatrozid

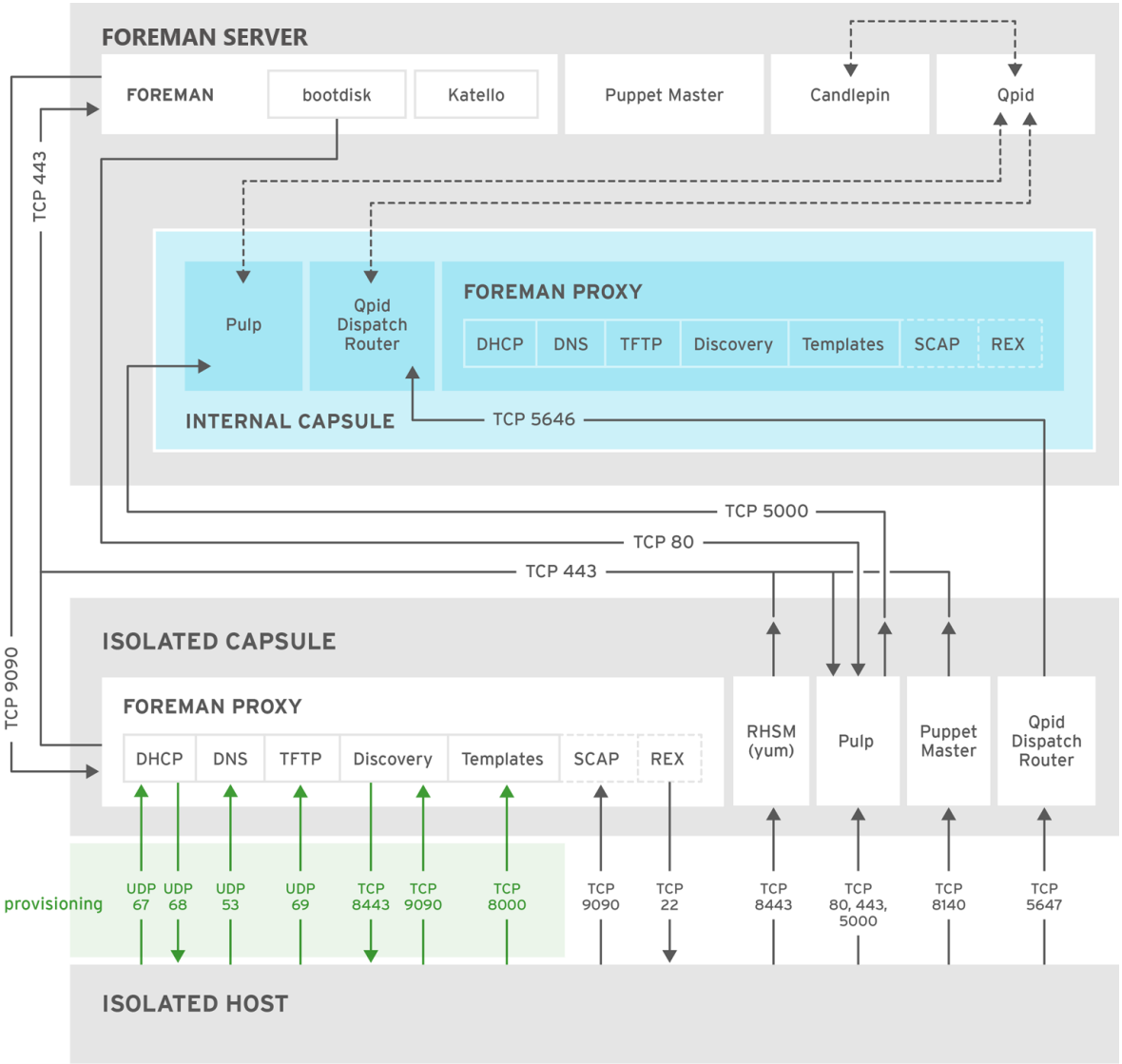
3.5. Konfiguracija SELinux portova

Iste portove koje smo propustili na vatrozidu potrebno je propustiti za SELinux. Svaki servis koji naknadno uključimo također je potrebno dodati u SELinux. SELinux je sistemska komponenta integrirana u Linux kernel, a definira pristup i prava svakog korisnika, aplikacije, procesa i drugih objekata na sustavu. Nadzire interakcije između navedenih objekata sigurnosnim politikama. U svakodnevnom radu običan korisnik gotovo da ni ne može primijetiti postojanje SELinuxa, no sistemski administratori bi se trebali pobrinuti za sigurnosne politike poslužitelja prema okolinama u kojima se oni nalaze. Naredbe koje sam koristio za konfiguriranje SELinuxa su slijedeće:

```
semanage port -a -t http_port_t -p tcp 80
semanage port -a -t http_port_t -p tcp 443
semanage port -a -t ssh_port_t -p tcp 22
semanage port -a -t puppet_port_t -p tcp 8140
semanage port -a -t amqp_port_t -p tcp 5647
semanage port -a -t commplex_main_port_t -p tcp 5000
```

Kod 3.8 Konfiguracija SELinux portova

Slika 3.2 zornije prikazuje koje portove treba propustiti kroz vatrozid te ih podesitit u SELinuxu. Ja nisam morao podesiti sve portove zbog toga što poslužitelji koje sam nadzirao ne koristi dinamički dodijeljene IP adrese te ne koristi DNS unutar Foremana već onaj unutar organizacije.



Slika 3.2 Foreman logički prikaz portova (izvor www.redhat.com)

3.6. Izrada vlastitog SSL certifikata

Zbog kompleksnosti Foremana preporučljivo je, a po iskustvu i obavezno napraviti certifikat prije same instalacije te ga ukomponirati u proces instalacije kako poslije ne bi imali problema s drugim servisima zbog razlika u certifikatima. Ja sam koristio OpenSSL kako bi generirao certifikat i ključ te sam ga poslao na potpisivanje kod organizacijskog certifikacijskog autoriteta. Kako prilično često izrađujem certifikate izradio sam datoteku sa svim potrebnim informacijama za certifikat te sam tu datoteku predao u naredbu za izradu certifikata. Ovo je sadržaj korištene datoteke naziva **openssl.cnf**:

```
[ req ]
default_bits = 2048
distinguished_name = req_distinguished_name
req_extensions = req_ext
prompt = no
[ req_distinguished_name ]
countryName = HR
stateOrProvinceName = Hrvatska
localityName = Zagreb
organizationName = ORG IT
commonName = foreman.prod.org.local
[ req_ext ]
subjectAltName = @alt_names
[alt_names]
DNS.1 = foreman.prod.org.local
```

Kod 3.9 Konfiguracijska datoteka za SSL certifikat

Kako bi izradio sam certifikat te ključ, pokrenuo sam slijedeću naredbu kojoj sam predao konfiguracijsku datoteku:

```
openssl req -out foreman.prod.org.local.csr -newkey rsa:2048 -nodes -keyout foreman.prod.org.local.key -config openssl.cn
```

4. Instalacija Foremana

Ako smo napravili dobru pripremu te kada smo sigurni kako želimo da izgleda naša arhitektura, instalacija samog Foremana je prilično trivijalan zadatak. Potrebno je dodati nekoliko repozitorija s kojih će instalacija moći povlačiti potrebne pakete, provjeriti certifikate te ukomponirati certifikate u instalacijski proces. Nakon toga potrebno je omogućiti Foremanu pristup do, u mojem slučaju VMware poslužitelja kako bi mogao komunicirati s hipervizorima koji nadgledaju virtualne poslužitelje. Naknadno je potrebno instalirati dodatke koji omogućavaju centralno udaljeno izvršavanje naredbi.

4.1. Dodavanje Red Hat i ostalih repozitorija

Kako bi uopće mogli započeti instalaciju potrebno je dodati repozitorije sa kojih će instalacija Foremana povlačiti pakete. Postoji i drugi slučaj instalacije kada nemamo pristup Internetu, u tom slučaju moramo preuzeti sve pakete ručno. Kada sam tek počeo sa instalacijom Foremana zbog sigurnosnih politika morao sam pokušati instalirati Foreman na taj način, no nakon višednevnog traženja svih paketa te zavisnosti jednih paketa o drugima shvatili smo da ćemo potrošiti previše vremena zbog jako velikog broja paketa. Drugi način je imati ograničen pristup Internetu, odnosno pristup samo odabranim repozitorijma. Kako bi se moglo doći do Red Hat-ovih repozitorija potrebno je imati otvoren račun na njihovoj stranici te registrirati poslužitelj ovom naredbom `subscription-manager register --auto-attach` nakon koje unesemo svoje korisničko ime i lozinku te vidimo sve dozvoljene repozitorije.

Kako bi mogli upravljati repozitorijima i paketima potrebno je instalirati yum-utils, te nakon toga podesiti samo potrebne Red Hat repozitorije:

```
yum-config-manager --disable "*"
yum-config-manager --enable rhel-7-server-rpms
yum-config-manager --enable rhel-7-server-optional-rpms
yum-config-manager --enable rhel-7-server-extras-rpms
```

Kod 4.1 Omogućavanje Red Hat repozitorija

Nakon Red Hat repozitorija moramo instalirati i ostale potrebne repozitorije:

```
yum -y localinstall
https://fedorapeople.org/groups/katello/releases/yum/3.5/kate
llo/el7/x86_64/katello-repos-latest.rpm
yum -y localinstall
https://yum.theforeman.org/releases/1.16/el7/x86_64/foreman-
release.rpm
yum -y localinstall https://yum.puppetlabs.com/puppetlabs-
release-pc1-el-7.noarch.rpm
yum -y localinstall
https://dl.fedoraproject.org/pub/epel/epel-release-latest-
7.noarch.rpm
```

Kod 4.2 Instalacija repozitorija

Sada kada imamo omogućene sve repozitorije možemo instalirati potrebne aplikacije na koje će se povezivati Foreman tijekom sve instalacije te kako bi imali sve pakete na posljednjim verzijama radimo osvježavanje (engl. update).

```
yum -y install foreman-release-scl python-django --nogpgcheck
yum -y update
yum -y install katello
```

Kod 4.3 Instalacija potrebnih aplikacija

4.2. Instalacija Foreman-a

Instalaciju Foremana pokrećemo jednostavnom naredbom `foreman-installer --scenario katello`, no moguće je proslijediti i dodatne parametre kojima možemo usmjeravati instalaciju. Za listu svih mogućih parametara imamo datoteku `katello-answers.yaml` koju možemo pronaći na lokaciji `/etc/foreman-installer/scenarios.d`. Unutar te datoteke su zapisani svi parametri te ih se na tom mjestu može promijeniti jer se sama instalacija poziva na tu datoteku. Ja sam bio zadovoljan s većinom parametara, jedino što sam htio promijeniti je SSL certifikat za web sučelje te druge servise. Zbog toga što je arhitektura Foremana kompleksna te postoji mnogo servisa koji se vežu jedan na drugog i na lanac certifikata, Foreman ima alat koji nam pomaže posložiti potrebne certifikate već prilikom instalacije. Taj alat je zapravo obična skripta

naziva `katello-certs-check` kojoj pridružujemo certifikate prema zastavicama, a ona nam generira naredbu za instalaciju. Ovom naredbom pozivamo skriptu:

```
katello-certs-check -c foreman.prod.org.local.cer -k
foreman.prod.org.local.key -r foreman.prod.org.local.csr -b
Org_Internal_CA_chain.crt
```

Kod 4.4 Pokretanje katello-certs-check skripte

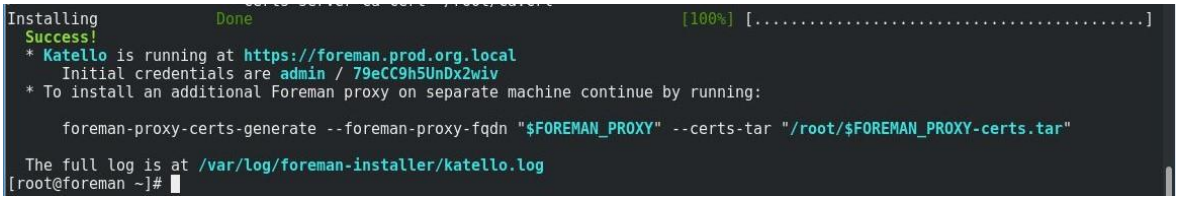
Zastavici **-c** pridružujemo potpisani certifikat od strane CA, kod ***-k*** stavljamo privatni ključ koji smo generirali OpenSSL-om, **-r** je certifikat na temelju kojeg smo dobili potpisani certifikat te na kraju zastavici **-b** stavljamo certifikat našeg autoritativnog poslužitelja.

Nakon što pokrenemo gore navedenu naredbu, kao izlaz dobijemo nekoliko opcija za pokretanje instalacije. Izabiremo onu koja odgovara našem sustavu, no još jednom napominjem da je najbolje to raditi prilikom čiste instalacije Foremana te sam tu opciju i sam odabrao. Ovom naredbom sam pokrenu instalaciju:

```
foreman-installer --scenario katello --certs-server-cert
"/root/foreman.prod.org.local.cer" --certs-server-cert-req
"/root/foreman.prod.org.local.csr" --certs-server-key
"/root/foreman.prod.org.local.key" --certs-server-ca-cert
"/root/Org_Internal_CA_chain.crt" --katello-proxy-url
http://proxy.prod.org.local --katello-proxy-port 8080
```

Kod 4.5 Naredba za instalaciju Foremana

Po uspješnoj instalaciji prikazuju se informacije za pristup web sučelju Foremana, administratorsko ime i lozinka te dodatni parametri za *proxy* i Puppet. Slika 4.1 prikazuje moju uspješnu instalaciju Foremana.



Slika 4.1 Uspješna instalacija Foremana

4.3. Instalacija Virt-Who agenta

Nakon same instalacije Foremana, potrebno je instalirati nekoliko dodataka te Virt-Who aplikaciju. Virt-Who je agent koji služi za izvještavanje o virtualnim poslužiteljima prema Foremanu. Ja sam Virt-Who povezao sa VMWare ESX, a za to mi je bilo potrebno da se na VMWare-u napravi korisnički račun koji će moći pregledati samo *UUID* virtualnih poslužitelja i hipervizora. Kako bi započeli instalaciju potrebno je pokrenuti naredbu `yum install virt-who` te po završetku instalacije iskoristiti naredbu `virt-who-password` kako bi generirali *hash* iz naše lozinke jer će ta ista lozinka biti korištena u konfiguracijskoj datoteci virt-who-a. Nakon što smo generirali lozinku potrebno je otići na lokaciju virt-who agenta, `/etc/virt-who.d/config.conf`, te ispuniti sljedeća polja sa potrebnim informacija:

```
[config]
type=esx
server=vcenter.prod.org.local
username=virtwhousr
encrypted_password= 448e4ed351161392a7678e68341a5ce3
owner=Org-IT
env=Library
hypervisor_id=hostname
```

Kod 4.6 Konfiguracija Virt-Who datoteke

Nakon što smo postavili datoteku potrebno je ponovo pokrenuti virt-who servis te pratiti logove koji nam mogu pokazati je li virt-who uspješno spojen sa VMWare-om te je li otkrio virtualne poslužitelje. Slika 4.2 pokazuje broj virtualnih poslužitelja te hipervizora koje je virt-who otkrio na VMware-u.

```
● virt-who.service - Daemon for reporting virtual guest IDs to subscription-manager
   Loaded: loaded (/usr/lib/systemd/system/virt-who.service; enabled; vendor preset: disabled)
   Active: active (running) since Tue 2018-04-24 18:59:37 CEST; 39s ago
 Main PID: 51549 (virt-who)
   CGroup: /system.slice/virt-who.service
           └─51549 /usr/bin/python /usr/bin/virt-who

Apr 24 18:59:35 vlp032.gzaop.local systemd[1]: Starting Daemon for reporting virtual guest IDs to
Apr 24 18:59:37 vlp032.gzaop.local /usr/bin/virt-who[51549]: [INFO] @main.py:183 - Using configu
Apr 24 18:59:37 vlp032.gzaop.local /usr/bin/virt-who[51549]: [INFO] @main.py:185 - Using reporte
Apr 24 18:59:37 vlp032.gzaop.local systemd[1]: Started Daemon for reporting virtual guest IDs to
Apr 24 18:59:40 vlp032.gzaop.local /usr/bin/virt-who[51549]: [INFO] @virt.py:885 - Report for co
Apr 24 18:59:41 vlp032.gzaop.local /usr/bin/virt-who[51549]: [INFO] @subscriptionmanager.py:202 -
35 hypervisors and 719 guests found
Apr 24 18:59:57 vlp032.gzaop.local /usr/bin/virt-who[51549]: [INFO] @subscriptionmanager.py:259 -
```

Slika 4.2 Status virt-who agenta

4.4. Instalacija Hammer CLI dodatka

Iako Foreman ima vlastito web grafičko sučelje, za obradu veće količine podataka odnosno za skriptiranje poslova može se koristiti Hammer CLI dodatak. Ja sam Hammer CLI koristio najviše kod promoviranja različitih CV-ova po okolinama organizacije iako kod takvih poslova treba uzeti u obzir da promoviranje radi velik broj pisanja po disku te troši velik broj inode-ova. Za instalaciju Hammer CLI i popratnih Foreman i Katello dodataka koji nude više opcija koristio sam ove naredbe:

```
yum install rubygem-hammer_cli
yum install rubygem-hammer_cli_foreman
yum install rubygem-hammer_cli_katello
```

Kod 4.7 Instalacija Hammer CLI dodatka

Kako bi pokrenuli neku od brojnih Hammer naredbi potrebno je u linijskom sučelju upisati hammer te neku od željenih naredbi i na kraju staviti zastavicu **-u** te **-p** nakon kojih treba unijeti ime korisnika te lozinku u čistom tekstu. Kako to nije najpraktičnije rješenje te pogotovo nije sigurno sa sigurnosnog stajališta, potrebno je napraviti konfiguracijsku datoteku u koju upišemo jednom korisničko ime i lozinku te se hammer cli poziva na tu datoteku. Kako je lozinka u čistom tekstu u toj datoteci najbolje i najsigurnije rješenje je da sakrijemo samu putanju do datoteke. Tako sam ja napravio datoteku na lokaciji `~/.foreman/cli-config.yml` te sam u nju dodao slijedeću konfiguraciju:

```
:modules:
    - hammer_cli_foreman

:foreman:
    :host: 'https://foreman.prod.org.local/'
    :username: 'admin'
    :password: 'lozinka'
```

Kod 4.8 Hammer CLI konfiguracijska datoteka

Konfiguracijska datoteka je veoma jednostavna jer moramo unijeti samo na koji modul se referenciramo, poslužitelj koji koristimo te korisničko ime i lozinku.

4.5. Instalacija Remote Execution dodatka

Ovaj dodatak omogućuje Foremanu izvršavanje naredbi na daljinu. Udaljeno izvršavanje (engl. Remote Execution) radi preko SSH servisa na portu 22 te smo zbog toga morali propustiti taj port na vatrozidu te SELinuxu. Moguće je koristiti naredbe na više od jednom poslužitelju te se mogu koristiti varijable kako bi se naredbe prilagodile našem okruženju. Ja sam se odlučio na instalaciju ovog dodatka jer mi je omogućio da pokrećem naredbe, instaliram zakrpe ili potrebne pakete na više poslužitelja odjednom. Za instalaciju ovog dodatka potrebno je pokrenuti slijedeće naredbe:

```
yum install tfm-rubygem-foreman_remote_execution \
            tfm-rubygem-foreman_remote_execution_core \
            tfm-rubygem-foreman-tasks \
            tfm-rubygem-foreman-tasks-core \
            rubygem-smart_proxy_dynflow \
            tfm-rubygem-smart_proxy_dynflow_core \
            rubygem-smart_proxy_remote_execution_ssh
```

Kod 4.9 Instalacija Remote Execution dodatka

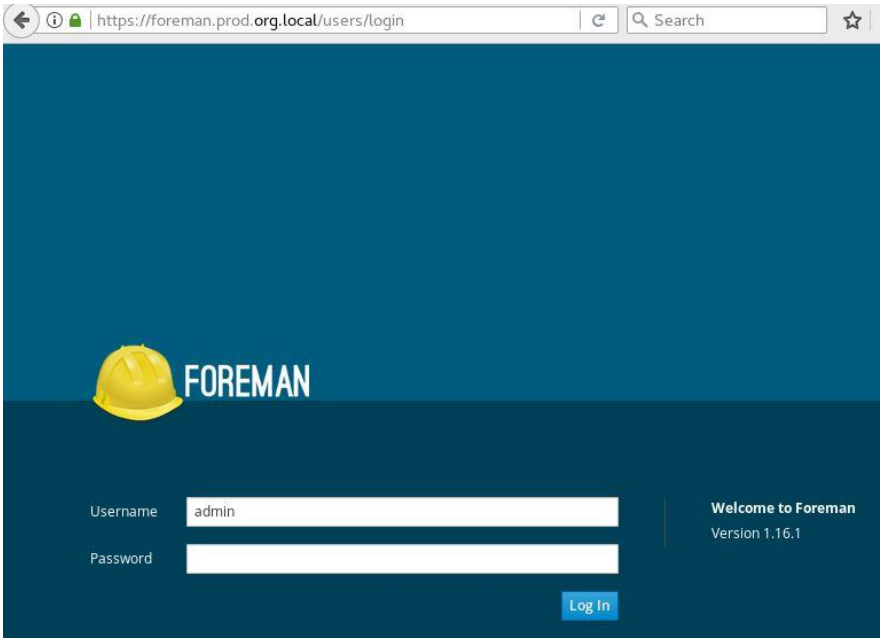
Nakon instalacije potrebno je razmijeniti SSH ključeve između Foremana i drugih virtualnih poslužitelja. Taj proces nije dugotrajan za mali broj poslužitelja, no za veći broj taj proces bi se morao ubrzati. Ja nisam morao razmjenjivat SSH ključeve na običan način već sam koristio Red Hat-ov upravitelj elektroničkim identitetima (engl. Identity Manager, skraćeno IDM). Nakon potrebne instalacije na IDM-u, otišao sam u postavke Foremana i stavio sam korisnika kojeg hoću koristiti za Remote Execution.

5. Uređivanje Foremana prema okruženju organizacije

Nakon iscrpne pripreme te instalacije vrijeme je da uredimo Foreman prema okruženju organizacije. U ovom dijelu potrebno je poznavati svoju organizaciju kako bismo mogli napraviti optimalnu konfiguraciju životnih okolina, potrebno je znati koje aplikacije koriste naši poslužitelji kako bi mogli posložiti ispravno repozitorije paketa. Također je potrebno promisliti o načinu instalacije zakrpi jer ćemo u sljedećim koracima stvarati kolekcije virtualnih poslužitelja te ćemo automatizirati taj proces. No prije nego što krenemo u izradu životnih okolina i ostalih zadataka potrebno je napraviti inicijalnu konfiguraciju.

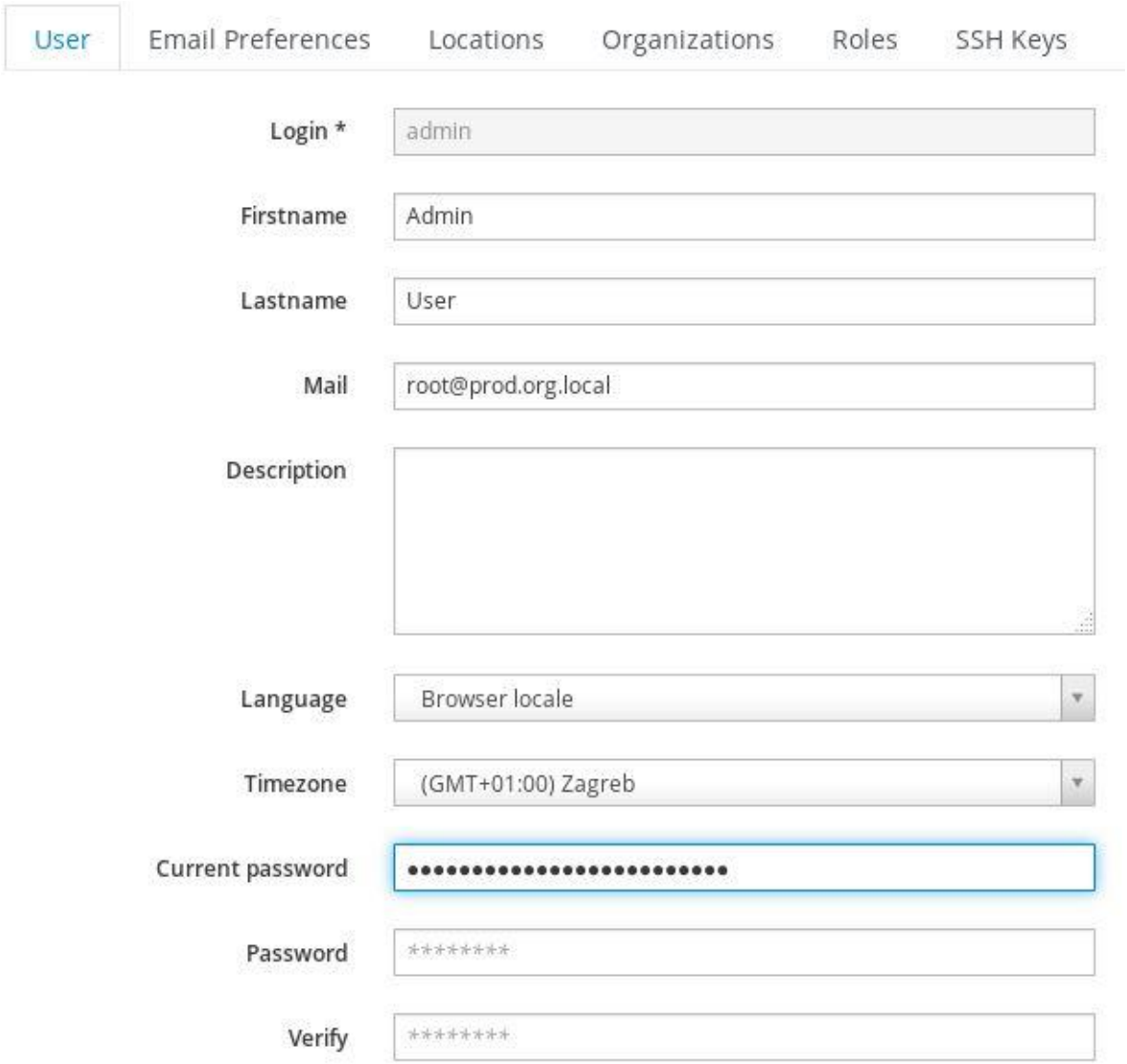
5.1. Inicijalna konfiguracija Foremana

Kako bi uopće mogli pristupiti Foremanu potrebno je zapamtiti poveznicu do Foremana te inicijalno korisničko ime i lozinku administratora. Te informacije dobijemo nakon uspješne instalacije, baš kao na Slika 4.1. Slijedeći poveznicu dolazimo do početne stranice Foremana na kojoj odmah možemo uočiti da naš SSL certifikat radi jer ne dobivamo nikakve greške te je potrebno unijeti korisničko ime i lozinku za pristup aplikaciji. Slika 5.1 prikazuje početnu stranicu.



Slika 5.1 Početna stranica Foremana

Nakon uspješne autentikacije kao korisnik s administratorskim ovlastima vidimo sve glavne izbornike koji su dostupni u Foremanu. Zbog sigurnosti prvo trebamo promijeniti administratorsku lozinku koju je generirao Foreman, a to se radi tako da otvorimo izbornik u gornjem desnom kutu gdje piše *Admin User* te odaberemo *My Account.* Slika 5.2 prikazuje sljedeći izbornik koji nam se otvara te u tom prozoru možemo promijeniti lozinku. U tom izborniku moramo odmah promijeniti i vremensku zonu kako kasnije u radu ne bi imali problema, konkretno kod rasporeda instalacije zakrpi.



Slika 5.2 Promjena lozinke i vremenske zone korisnika

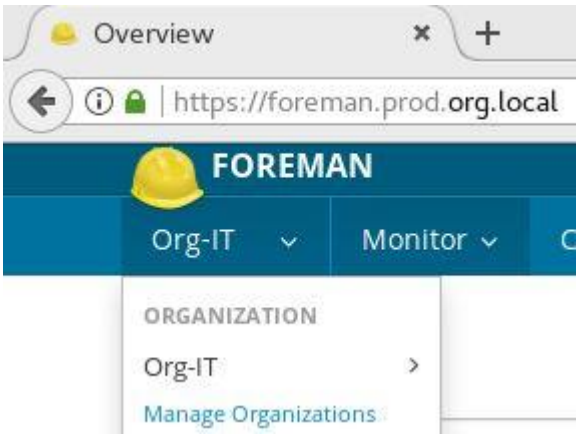
5.2. Izrada organizacije

Nakon što smo napravili inicijalnu konfiguraciju potrebno je izraditi organizaciju jer ćemo većinu sljedećih koraka povezivati s organizacijom kojoj pripadamo. Sama logika organizacija je da jedan korisnik može administrirati nekoliko organizacija s jedne centralne lokacije. Na početku svi pripadaju *Default Organization* organizaciji i prikladno je da tu organizaciju promijenimo i stavimo ime svoje organizacije. Ta promjena se kao i većina ostalih može napraviti preko grafičkog sučelja ili preko hammer cli naredbi. Ja sam zbog prijašnjih testiranja koristio oba načina no ovdje ću iskoristiti naredbe zbog preglednosti teksta.

```
hammer organization create --name "Org-IT" --label "Org-IT" -
-description "Ovo je naša organizacija"
```

Kod 5.1 Izrada organizacije

Nakon što smo unijeli gornju naredbu možemo kliknuti na izbornik kod organizacije i odabrati našu organizaciju, kao što prikazuje Slika 5.3.

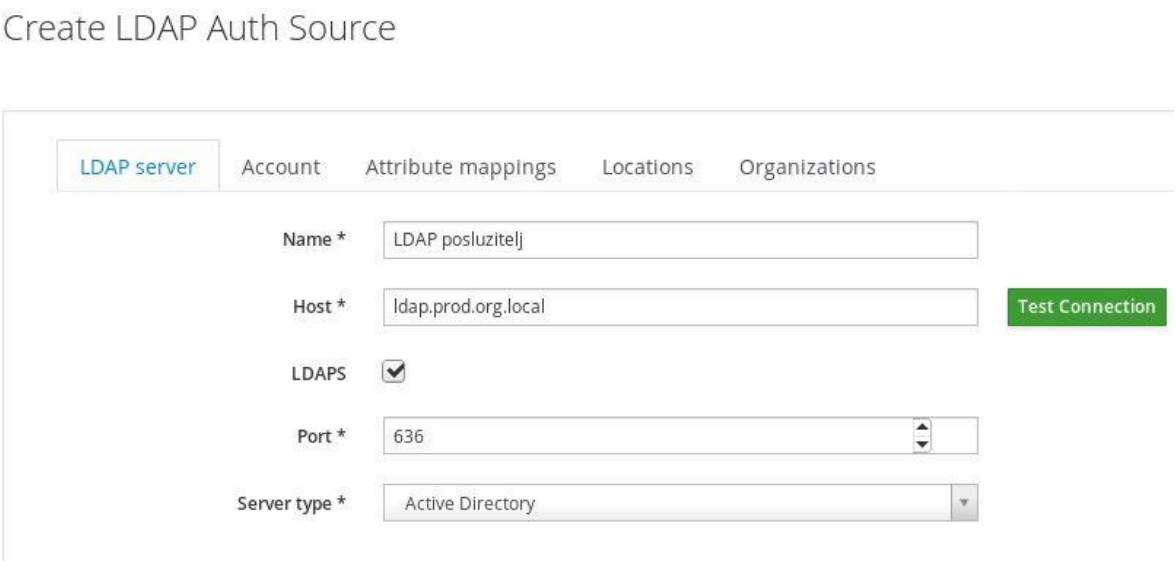


Slika 5.3 Odabir izrađene organizacije

5.3. Povezivanje s LDAP poslužiteljem

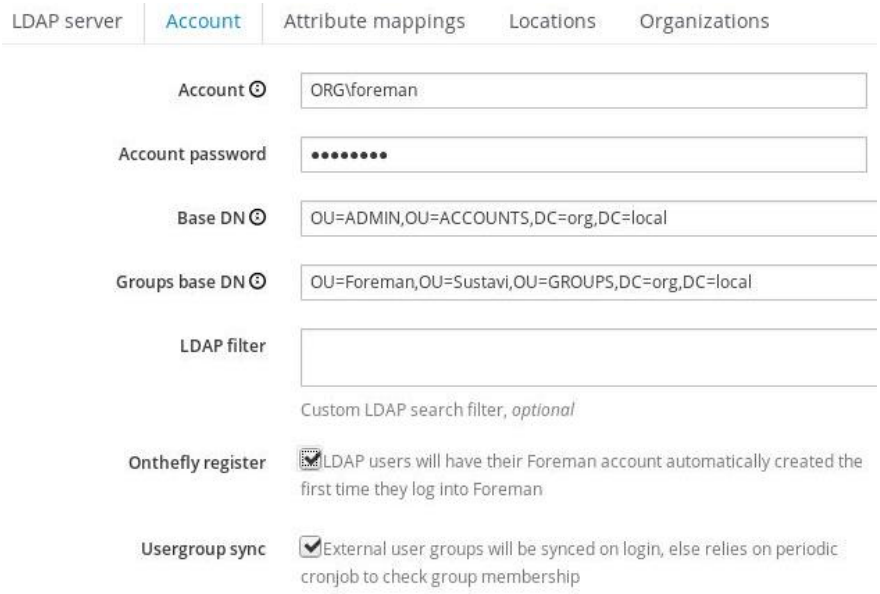
Kako bismo prestali koristiti administratorski račun i kako ne bismo morali izrađivati nove korisničke račune te im dodjeljivati prava, moguće je Foreman povezati s LDAP poslužiteljem. Tako će Foreman moći pročitati i iskoristi sva ona ista prava za svakog korisnika u LDAP-u. Kako bi započeli povezivanje potrebno je otvoriti izbornik *Administer* te odabrati *LDAP Authentication*. Otvoriti će se novi prozor koji će nam ponuditi da stvorimo

novi autentikacijski izvor odnosno da povežemo naš LDAP poslužitelj. Nakon što otvorimo poveznicu na stvaranje novog izbora, otvara se stranica sa nekoliko izbornika gdje trebamo popuniti obavezna polja. Prvi izbornik je LDAP Server u kojem trebamo napraviti ime za identifikaciju, ime samog LDAP poslužitelja, označiti hoćemo li koristiti sigurniju verziju LDAP-a te koji tip servera je naš LDAP poslužitelj. Slika 5.4 prikazuje moju konfiguraciju.



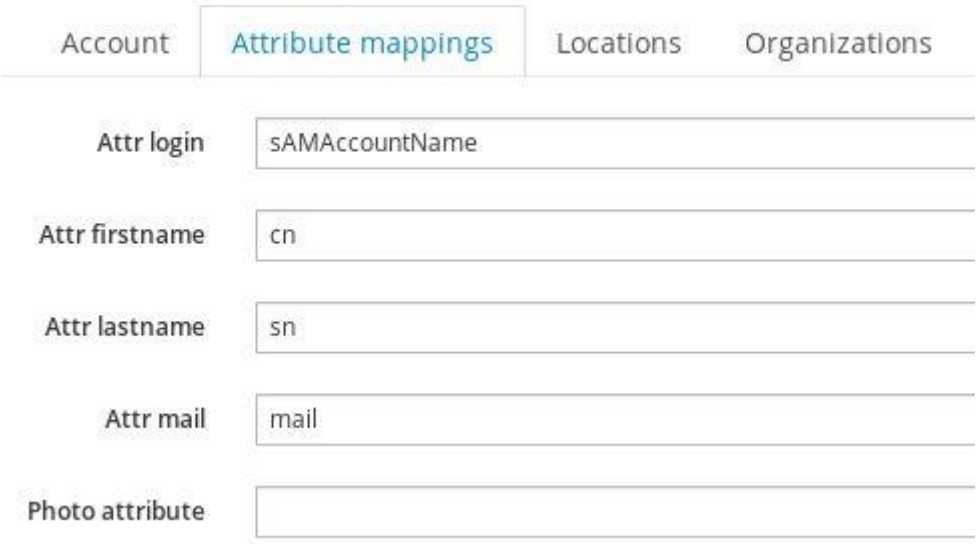
Slika 5.4 Konfiguracija LDAP poslužitelja

Nakon što smo popunili sva polja u *LDAP Server-u*, potrebno je napraviti isto u *Account* izborniku gdje navodimo kojim korisnikom se povezujemo na LDAP poslužitelj te DN zapise. Slika 5.5 prikazuje moju konfiguraciju.



Slika 5.5 Konfiguracija korisničkog računa za pristup LDAP-u

Zadnje što nam preostaje za konfigurirati su atributi na koje će se LDAP povezati. Slika 5.6 prikazuje konfiguraciju, kako imamo samo jednu organizaciju i lokaciju te izbornike ne trebamo popunjavati.



Slika 5.6 Konfiguracija LDAP atributa

5.4. Izrada životnih okolina organizacije

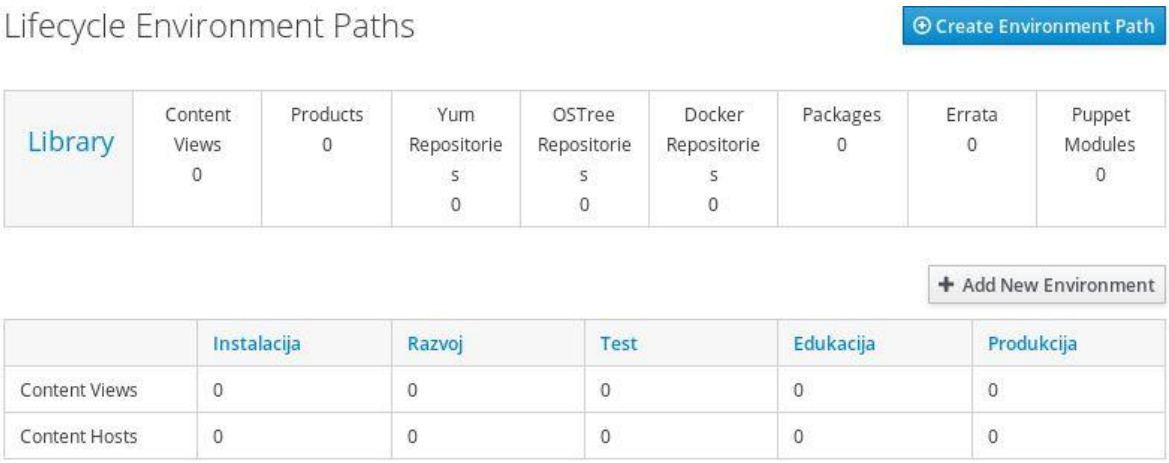
Kod izrade životnih okolina važno je konzultirati se sa razvojnim timom koji razvija aplikacije kako bi znali koliko životnih okolina oni koriste u razvoju aplikacije. Prema tome možemo u Foremanu stvoriti životnu stazu (engl. Environment path) po kojoj ćemo promovirati sadržaj za razvoj aplikacija. U mojem slučaju razvojni tim koristi pet okolina kroz koje oni testiraju aplikacije. Te okoline su, instalacija, razvoj, test, edukacija i produkcija. Kako bi izradili u Foremanu životne okoline potrebno je ići u izbornik *Content* te izabrati *Lifecycle Environments* i odabrati *Create Environment Path* kako bi počeli stvarati stazu životnih okolina. To je proces kroz grafičko sučelje, no ja sam to napravio kroz konzolno sučelje koristeći hammer cli:

```
hammer lifecycle-environment create --name \
"Instalacija" --label "Instalacija" --description "Ovo je
instalacijska okolina" --prior-id 2 --organization-label
"Org-IT"
hammer lifecycle-environment create --name \
"Razvoj" --label "Razvoj" --description "Ovo je razvojna
okolina" --prior-id 3 --organization-label "Org-IT"
```

```
hammer lifecycle-environment create --name \
"Test" --label "Test" --description "Ovo je testna okolina" -
-prior-id 4 --organization-label "Org-IT"
hammer lifecycle-environment create --name \
"Edukacija" --label "Edukacija" --description "Ovo je
edukacijska okolina" --prior-id 5 --organization-label "Org-
IT"
hammer lifecycle-environment create --name \
"Produkcija" --label "Produkcija" --description "Ovo je
produkcijska okolina" --prior-id 6 --organization-label "Org-
IT"
```

Kod 5.2 Izrada životnih okolina organizacije

Ako osvježimo *Lifecycle Environment Path* vidjet ćemo izrađene okoline kao na Slika 5.7.



Slika 5.7 Životne okoline organizacije

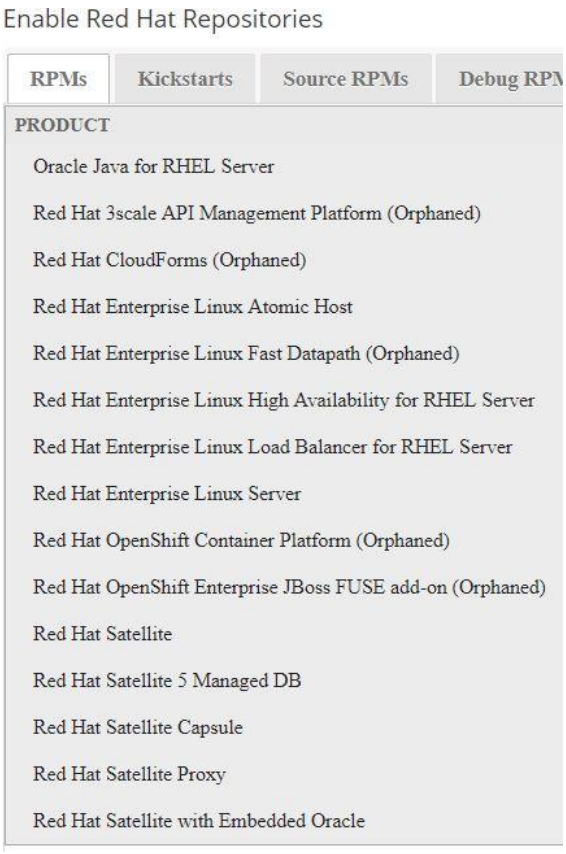
5.5. Izrada repozitorija

Došli smo na jednu od glavne ideje ovog rada, a to je izrada repozitorija, odnosno centralnog repozitorija koji će moći opsluživati sve poslužitelje kada trebaju neki od paketa. U slučaju kada nemamo centralni repozitorij postoje dvije opcije kako preuzeti paket za instalaciju aplikacije, jedna je da omogućimo svim poslužiteljima izlaz na Internet, a druga je da s nekog trećeg poslužitelja ili računala preuzimamo pakete i premještamo ih na željeni poslužitelj. Obje opcije imaju nedostataka u vidu sigurnosti te same praktičnosti. Kod centralnog repozitorija imamo samo jedan poslužitelj koji ima pristup Internetu, taj isti poslužitelj kod sebe drži sve potrebne repozitorije te kada neki od poslužitelja zatreba neki paket, vrlo lako ga može zatražiti od Foremana te ako je taj repozitorij omogućen tom poslužitelju paket se

počinje instalirati zajedno sa svim ovisnostima (engl. dependency). U uvodu sam napisao kako Foreman radi i kako dopušta poslužiteljima pristup repozitorijima, ovdje ću objasniti kako izraditi repozitorije.

5.5.1. Izrada Red Hat repozitorija

U slučaju kada imamo zakupljenu licencu od strane Red Hat-a te imamo pristup službenim repozitorijima, prvo moramo na Red Hat korisničkom portalu kreirati manifest koji će sadržavati licence te taj manifest preuzeti na računalo i uvesti ga u Foreman. Kako bi uvezli manifest u Foreman potrebno je otići u izbornik *Content* i tamo izabrati *Red Hat Subscriptions*. Ako nemamo ni jedan manifest uvezen odmah nam se otvara prozor gdje možemo kliknuti na gumb pretraži (engl. Browse) te odabrati željeni manifest i kliknuti na gumb učitaj (engl. Upload). Nakon što učitavanje završi potrebno je ponovo otići u izbornik *Content* i ovaj put odabrati *Red Hat Repositories* gdje nam se prikažu svi službeni repozitoriji koji su pokriveni licencom te na tom mjestu možemo odabrati one koje želimo koristiti. Slika 5.8 prikazuje dostupne repozitorije koji su pokriveni licencom Virtual Data Center.

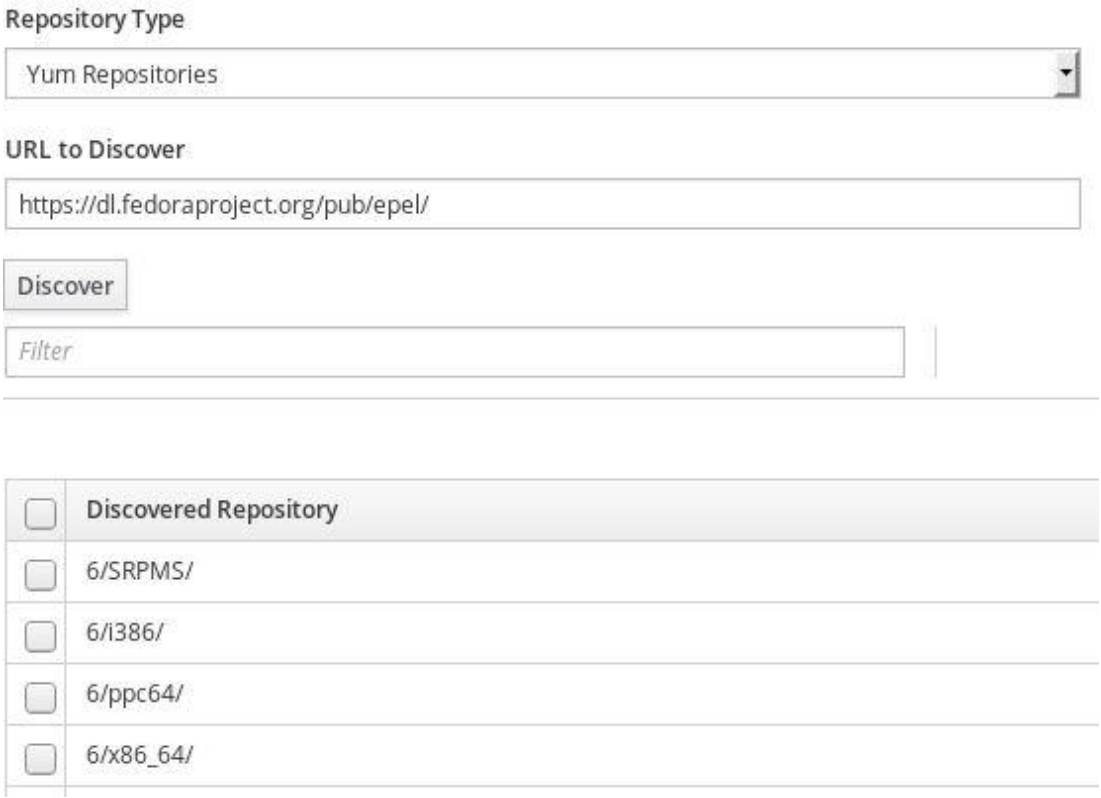


Slika 5.8 Red Hat repozitoriji

5.5.2. Izrada repozitorija trećih strana

U slučaju kada organizacija nema Red Hat repozitorije, može koristiti repozitorije trećih strana, odnosno službene repozitorije aplikacija koje koriste ili operativnih sustava kao što je CentOS. Postoje dva načina kako dodati repozitorije, jedan način je da koristimo *Repo Discovery* opciju kojoj proslijedimo poveznicu prema repozitoriju i ako taj servis pronađe repozitorije na toj poveznici odmah ih možemo odabrati. Ta opcija ne uspije uvijek pronaći repozitorije pa smo prisiljeni koristiti drugu opciju gdje trebamo znati apsolutnu putanju poveznice do željenog repozitorija.

Kada želimo izraditi repozitorij, a da nije Red Hat, moramo otvoriti izbornik *Content* i odabrat podizbornik *Products*. U gornjem desnom kutu tog prozora možemo kliknuti na gumb *Repo Discovery*. Tu opciju koristimo kada ne znamo apsolutnu putanju poveznice do repozitorija. Kada nam se otvori taj prozor izaberemo tip repozitorija koji želimo koristiti, u mojem slučaju to je yum repozitorij te kopiramo poveznicu u polje ispod i pritisnemo gumb otkrij (engl. Discover). U mojem slučaju kao što prikazuje Slika 5.9 tražimo Epel repozitorij, no nemamo apsolutnu putanju do repozitorija pa Foreman izlista sve repozitorije koje može pronaći na toj poveznici.

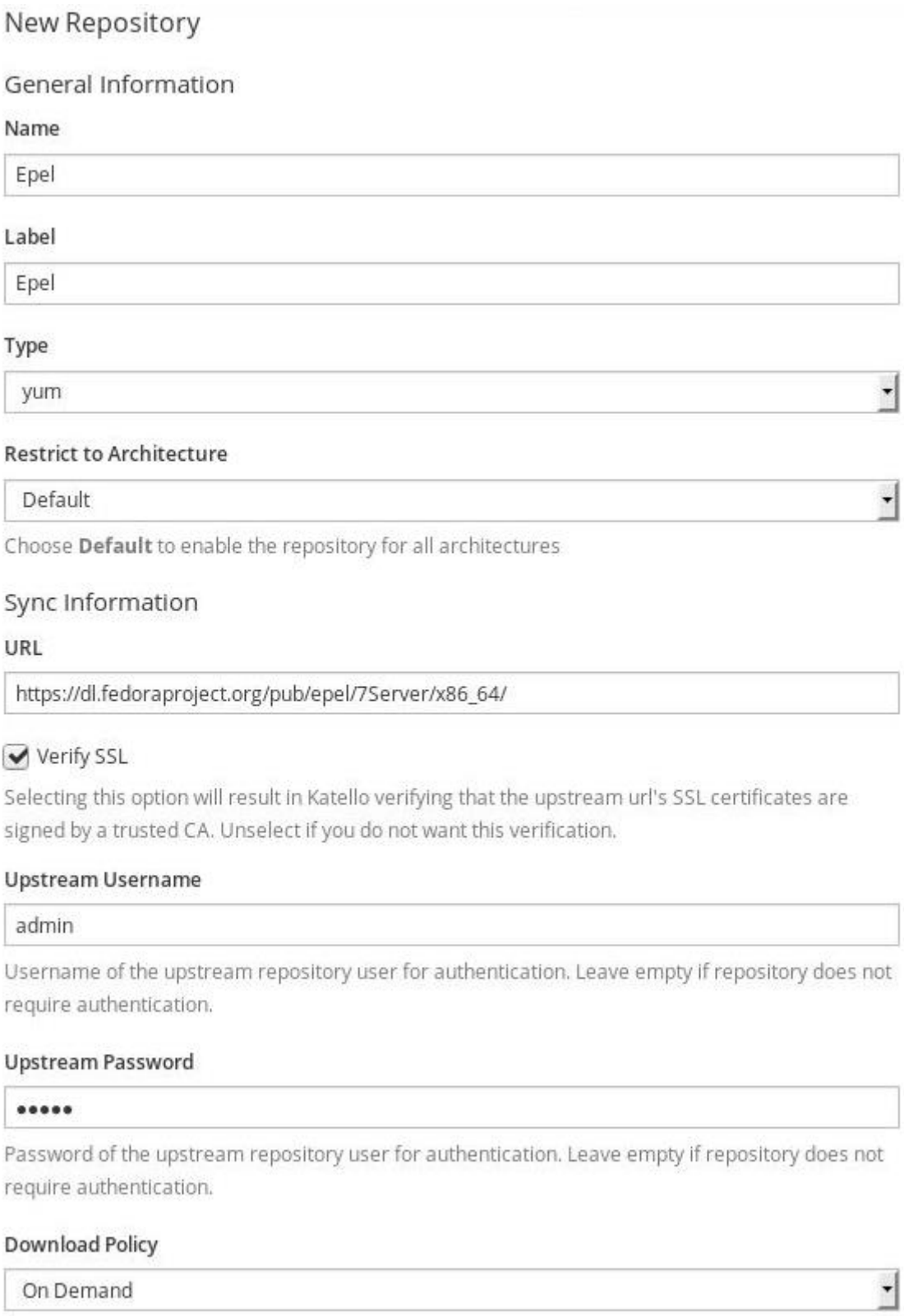


Slika 5.9 Otkrivanje repozitorija

Kada pronađemo repozitorij koji nam odgovara, označimo ga i odaberemo gumb *Create Selected* te u sljedećem prozoru po izboru napravimo novi proizvod (engl. Products) ili nadodamo taj repozitorij već postojećem proizvodu.

U drugom slučaju kada imamo apsolutnu putanju do željenog repozitorija možemo preskočiti otkrivanje istog i odmah krenuti u izrađivanje proizvoda. Moramo doći natrag do prozora *Products* gdje pritisnemo gumb *Create Product*. Nakon toga unesemo ime proizvoda, ako se radi o samo jednom repozitoriju onda možemo proizvod nazvati po tom repozitoriju, u slučaju kada trebamo više repozitorija za jednu aplikaciju tada možemo proizvod nazvati prema toj aplikaciji. Ja sam proizvod nazvao Epel i pritisnuo sam gumb spremi. Foreman nas pozicionira u taj proizvod na izbornik *Repositories* gdje možemo pritisnuti gumb *New Repository* i popuniti potrebna polja kao što je vidljivo na Slika 5.11. U polje ime upisujemo ime repozitorija, u polje tip odabiremo koji tip repozitorija želimo koristiti, možemo ograničiti repozitorij na samo jednu arhitekturu te najbitnije polje jest polje gdje upisujemo apsolutnu putanju poveznice. Još jedno zanimljivo polje je politika preuzimanja (engl. Download Policy) gdje možemo izabrati politiku između tri opcije. *On Demand* opcija nam nudi mogućnost da sinkroniziramo samo meta podatke, te da preuzme na Foreman paket tek kada ga neki drugi poslužitelj zatraži. Ovo je opcija koja se meni najviše sviđa i koju koristim zbog toga što ne opterećuje Internet vezu s preuzimanjem velikog broja paketa odjednom te ne trebam proširivati LVM prostor toliko često. Druge dvije opcije preuzimaju sve pakete u repozitoriju, *Immediate* opcija odmah po sinkronizaciji, a *Background* opcija nakon inicijalne sinkronizacije. Kada smo odabrali sve željene opcije pritisnemo gumb spremi te se vraćamo u *Repositories* prozor gdje vidimo novo izrađeni repozitorij. Taj repozitorij trebamo sinkronizirati kako bi mogli početi preuzimati pakete. To radimo tako da označimo repozitorij i odaberemo gumb *Sync Now*. Nakon sinkronizacije u *Content* skupini vidimo broj paketa u repozitoriji, u mojem slučaju vidljivo na Slika 5.10.

Slika 5.10 Broj paketa nakon sinkronizacije



Slika 5.11 Izrada repozitorija

5.6. Podešavanje rasporeda sinkronizacije repozitorija

Kako bi uvijek imali posljednje verzije paketa te sigurnosnih zakrpa i poboljšanja, potrebno je sinkronizirati repozitorije. Kako nema nekih preporuka koliko često bi se trebalo sinkronizirati, moja istraživanja su pokazala da to ovisi o veličini organizacije te broju repozitorija. Ja sam odlučio napraviti plan sinkronizacije tako da se svaku noć sinkroniziraju svi repozitoriji, a onda po potrebi ili prije ciklusa instaliranja zakrpi radim promoviranje tih novih repozitorija na sve CV-ove. Kako bi izradili plan sinkronizacije potrebno je otići u izbornik *Content* i odabrati *Sync Plans* te nakon što se otvori novi prozor kliknemo u gornji desni kut *Create Sync Plans*. Na sljedećem prozoru potrebno je ispuniti polja imena tog rasporeda koji izrađujemo, možemo mu staviti prikladan opis te izaberemo jednu od opcija intervala, što može biti svaki sat, dnevno ili tjedno te na kraju točno vrijeme u koje će sinkronizacija započeti. Kada smo to napravili potrebno je pritisnuti gumb spremi te na sljedećem prozoru odabrati koji proizvod ili više njih želimo sinkronizirati. Slika prikazuje moj plan sinkronizacije repozitorija.

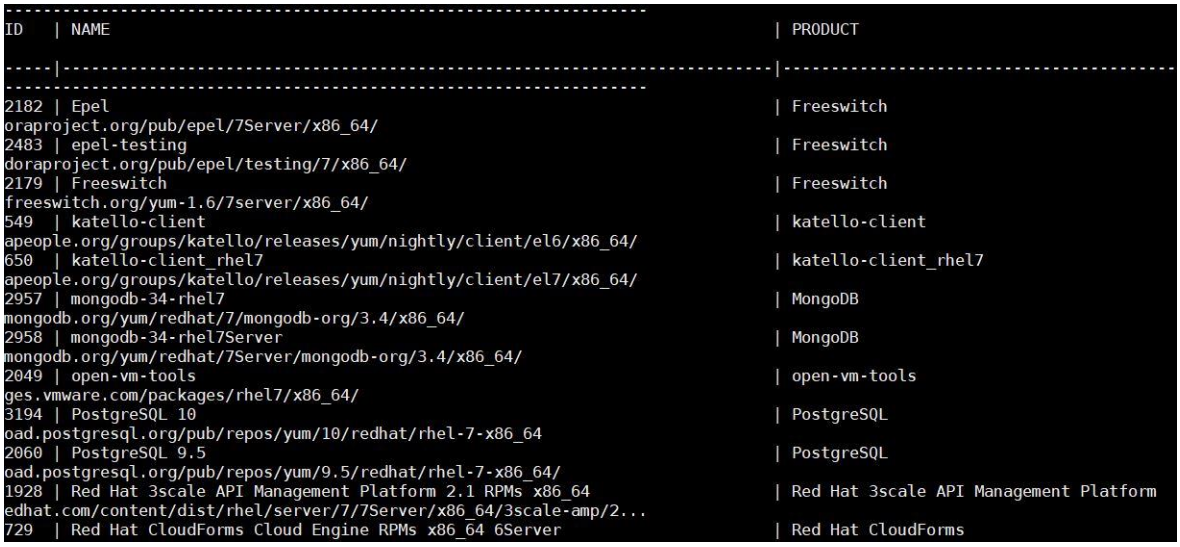
Name	Description	Sync Status	Repositories
PostgreSQL		Last synced about 8 hours ago.	2
Red Hat CloudForms		Last synced about 1 month ago.	2
Red Hat Enterprise Linux Fast Datapath		Last synced about 1 month ago.	1
Red Hat OpenShift Container Platform		Last synced about 1 month ago.	3

Slika 5.12 Plan sinkronizacije repozitorija

5.7. Izrada CV-a i kompozitnih CV-ova

U Foremanu koristimo CV-ove kako bi određenoj aplikaciji ili operacijskom sustavu dodijelili njihove repozitorije. Praktično to izgleda tako da napravimo poseban CV za Red Hat repozitorij, poseban CV za primjerice PostgreSQL bazu te kasnije možemo te CV-ove promovirati kroz životne cikluse aplikacija i dodjeljivati ih poslužiteljima. Produkcijska okolina može zahtijevati neke starije pakete dok razvojna uvijek zahtjeva najnovije pakete kako bi se sve moglo temeljito testirati. Ja sam u primjeru ispod za svaki repozitorij napravio

jedan CV. Izrada CV-a je moguća kroz web sučelje, no po iskustvu brža je izrada preko hammer cli konzole. Kako bi mogli izraditi CV potrebno je znati koje repozitorije imamo te njihove identifikacijske brojeve što možemo saznati ovom naredbom `hammer repository list --organization "ACME"`. Slika 5.13 prikazuje skraćeni popis repozitorija koji se nalaze na Foreman poslužitelju.



Slika 5.13 Popis repozitorija

Sada kad imamo identifikacijski broj repozitorija možemo krenuti u izradu CV te nakon što smo ga izradili potrebno je napraviti publiciranje i promocija u željenu životnu okolinu.

```
hammer content-view create \
--name "Epel" \
--description "Epel content view" \
--repository-ids 2182 \
--organization "Org-IT"
hammer content-view publish \
--name "Epel" \
--description "Inicijalni Epel content view" \
--organization "Org-IT"
hammer content-view version promote \
--content-view "Epel" \
--version 1 \
--to-lifecycle-environment "Instalacija" \
--organization "Org-IT"
```

Kod 5.3 Izrada, publiciranje i promocija CV-a

U kodu iznad ja sam izradio Epel CV, napravio sam publiciranje kako bi bio svima vidljiv te sam ga posebno promovirao u Instalacijsku okolinu kako bi ga samo ti poslužitelji mogli koristiti.

Ja sam u taj jedan CV mogao ugraditi više repozitorija, no time si smanjujem skalabilnost sustava. Moja ideja vodilja je bila imati što više komponenata na koje mogu utjecati i upravo zbog toga sam napravio jedan CV za svaki repozitorij. Tu naizgled postoji problem u toj logici jer što ako neka naša aplikacija zahtjeva pakete iz više repozitorija, svakom poslužitelju moguće je pridodati samo jedan CV što u ovom mojem slučaju nije moguće. Ovdje u priču ulaze kompozitni CV-ovi koji nadodaju još jedan sloj kompleksnosti sustavu no sa njima imamo potpuno skalabilan sustav. Različite aplikacije mogu zahtijevati više repozitorija, ali možda neke zahtijevaju novije paketa, a neke starije. U ovom slučaju na to isto možemo utjecati jer možemo u kompozitnim CV-ovima birati koju verziju CV-a želimo koristiti, a mi s druge strane možemo imati CV-ove raspodijeljene na više verzija u ovisnosti o verzijama paketa koje sadrže. Kako bi izradili kompozitne CV-ove potrebno je izlistati sve CV-ove i znati koji su njihovi identifikacijski brojevi, to možemo učiniti ovom naredbom `hammer content-view version list --organization "Org-IT"`. Slika 5.14 prikazuje izlistane CV-ove.

CONTENT VIEW ID	NAME	LABEL	COMPOSITE	LAST PUBLISHED	REPOSITORY IDS
32	3scale	3scale		2017/11/16 08:29:31	1928
2	Default Organization View	683aff8a-914d-4bd7-b623-ca417cbf0df5		2017/10/06 13:30:36	
38	Freeswitch	Freeswitch		2018/02/21 08:31:33	2179, 2182, 2483
33	Fuse	Fuse		2018/04/03 12:50:06	1972
17	katello-client	katello-client		2017/10/18 06:46:31	549
22	katello-client_rhel7	katello-client_rhel7		2018/03/01 15:02:55	650
43	MongoDB	MongoDB		2018/03/26 13:12:38	2957, 2958
23	Openshift	Openshift		2018/01/02 10:10:56	3, 8, 728, 729, 7
37	PostgreSQL	PostgreSQL		2018/04/12 13:48:52	3194, 2060
29	RHEL	RHEL		2018/04/19 12:15:52	549, 3, 4, 5, 6,

Slika 5.14 Lista CV-ova

Sama izrada kompozitnih CV-ova je vrlo slična izradi običnih CV-ova no kod ovih je potrebno dodati zastavicu `--composite` dok je publiciranje i promoviranje potpuno isto.

```
hammer content-view create \
--composite \
--name "Moodle" \
--description "CCV za Moodle sadrži OS repo i bazu" \
--component-ids 4,14 \
--organization "Org-IT"
```

Kod 5.4 Izrada kompozitnih CV-ova

5.8. Izrada aktivacijskih ključeva

Ako bismo sada htjeli staviti neki poslužitelj pod nadzor Foremana to bismo trebali učiniti tako da instaliramo Foreman paket, morali bismo ga ručno staviti u okruženje u kojem treba biti, dodijeliti mu ručno kompozitni CV te pretplatu (engl. subscription) kako bi mogao koristiti repozitorije. Tih par koraka su sami po sebi procedura koja troši vrijeme, a kada bi to morali napraviti na više stotina poslužitelja tada bi vjerojatno odustali od Foremana. Srećom postoji jednostavniji pristup stavljanja poslužitelja pod nadzor, a taj je koristeći aktivacijske ključeve. Kod aktivacijskih ključeva mi unaprijed zadajemo u koje životno okruženje stavljamo poslužitelj, koji kompozitni CV te pretplate treba imati. Nakon što to napravimo dobijemo kao rezultat poveznicu koju kao naredbu unesemo u poslužitelj te se on automatikom registrira. Prije nego što krenemo u proces izrade aktivacijskog ključa potrebno je izlistati ovom naredbom `hammer subscription list --organization „Org-IT“` sve pretplate koje su nastale izradom repozitorija i zapisati UUID željene pretplate. U slijedećem primjeru ja sam izradio aktivacijski ključ za Splunk poslužitelje u razvojnoj okolini te sam im dodijelio Splunk pretplatu.

```
hammer activation-key create \
--name "Splunk" \
--unlimited-content-hosts true \
--description "Aktivacijski kljuc za splunk posluzitelje u
razvoju" \
--lifecycle-environment "Razvoj" \
--content-view "Splunk"  \
--organization "Org-IT"
hammer activation-key add-subscription \
--name "Splunk" \
--subscription-id gg808181533518d50152354246e901bb \
--organization "Org-IT"
```

Kod 5.5 Izrada aktivacijskog ključa

5.9. Dodavanje virtualnih poslužitelja u Foreman

Sada kada imamo sve potrebne korake možemo krenuti dodavati poslužitelje u nadzor Foremana. Sve što je potrebno su dvije naredbe, prva je da poslužitelj od Foremana zatraži paket kojim će se registrirati na Foremana te da unese aktivacijski ključ kako bi automatski bio smješten u odgovarajuću okolinu s odgovarajućim pretplatama. Poveznica za instalirati paket je ista za sve poslužitelje, a možemo je naći u izborniku *Hosts*, pa odaberemo podizbornik *Content Hosts* i kliknemo na gumb *Register Content Host*. Foreman nam sam ponudi poveznicu te naredbu kojom instaliramo paket, `rpm -Uvh http://foreman.prod.org.local/pub/katello-ca-consumer-latest.noarch.rpm`. Nakon što se taj paket uspješno instalira možemo otići natrag u Foreman te u izbornik *Content* i podizbornik *Activation Keys* gdje pronađemo i otvorimo ključ koji smo izradili. Nakon što smo otvorili taj ključ vidjet ćemo poveznicu koju moramo kopirati te zalijepiti u novi poslužitelj. Ako je sve prošlo uspješno vidjet ćemo rezultat kao na Slika 5.15 te ćemo vidjeti taj poslužitelj u izborniku *Hosts*, podizborniku *Content Hosts*.

Slika 5.15 Uspješna registracija poslužitelja u Foreman

5.10. Izrada kolekcija poslužitelja

Kolekcija poslužitelja služe tome da lakše možemo izvoditi različite operacije nad više poslužitelja odjednom. Ja kolekcije poslužitelja koristim za upravljanje zakrpama, podijelim poslužitelje prema okolinama ili prema nekim zajedničkim atributima te tako napravim raspored automatskog instaliranja zakrpa. Za izradu koristim hammer cli naredbe:

```
hammer host-collection create \
--organization-label "Org-IT" \
--name "Instalacija_1"
hammer host-collection add-host \
--id 1 \
--host-ids posluzitelj1,posluzitelj2
```

Kod 5.6 Izrada kolekcija poslužitelja

6. Udaljeno upravljanje poslužitelja s centralne lokacije

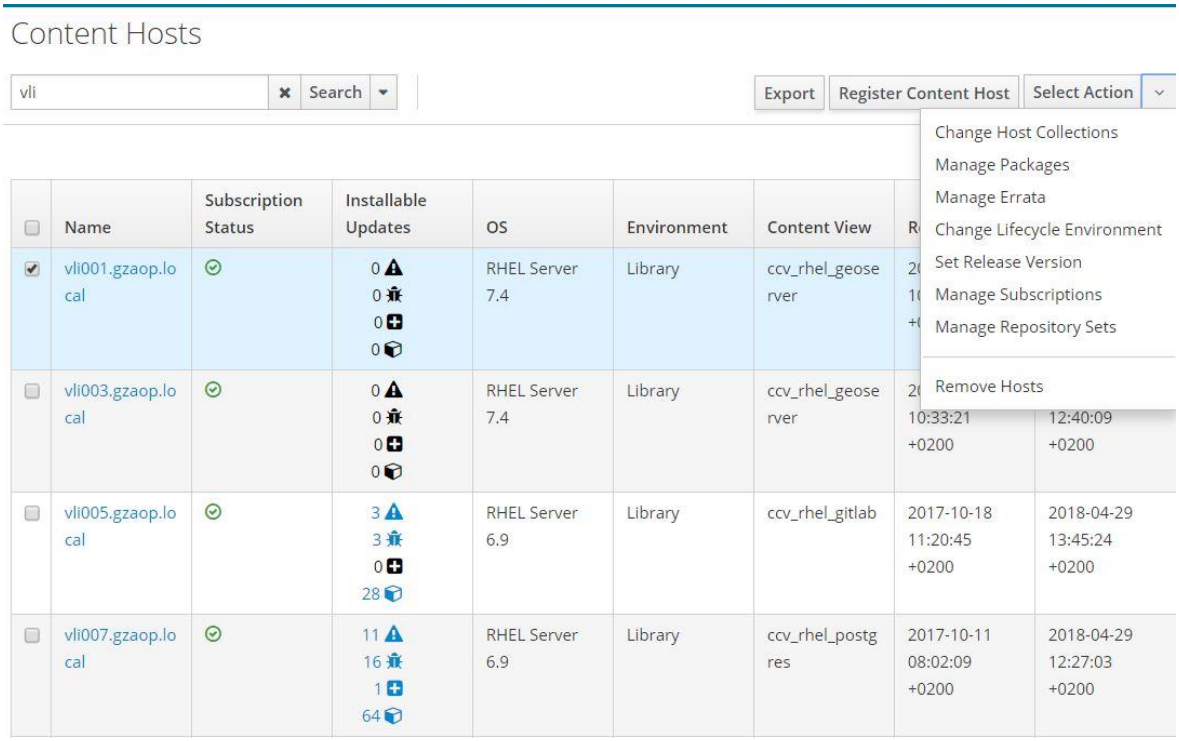
Sada kada smo izradili dobru pripremu te smo Foreman prilagodili našoj organizaciji moguće je upravljati svim poslužiteljima bez obzira koliko ih ima na udaljen način s jedne centralne lokacije. Ovakav pristup je u današnjem svijetu jedino moguć zbog količine poslužitelja koje treba pratiti i kojima treba upravljati. Ovakav način je u organizaciji u kojoj ja radim ubrzao upravljanje zakrpama preko 50%, odnosno tamo gdje je jedan inženjer trebao tri tjedna da ručno instalira zakrpe te provjeri rad računala, sada je potrebno tjedan i pola za sve poslužitelje u svim okolinama.

6.1. Udaljeno upravljanje paketima

Nakon što registriramo neki poslužitelj na Foreman, potrebno je ručno instalirati katello-agent paket koji nam omogućuje da Katello unutar Foremana može vidjeti trenutne verzije paketa te koji svi paketi imaju mogućnost osvježavanja i koje *Errate* treba instalirati. *Errata* je skupni naziv za sigurnosne pakete, *bug* popravke te pakete poboljšanja (engl. Security, Bug Fixes, Enhancements). Kako bismo pokrenuli upravljanje paketima moramo otići u izbornik *Hosts* te *Content Hosts* gdje vidimo listu svih poslužitelja koji su registrirani u Foreman. Tamo možemo označiti jedan ili više željenih poslužitelja te u gornjem desnom kutu odabrati *Select Action* te podizbornik *Manage Packages* kao što prikazuje Slika 6.2. Nakon toga pojavljuje se prozor u kojem treba upisati ime paketa te odabrati koju opciju, instalaciju, brisanje ili osvježavanje želimo. Kada odaberemo željenu opciju, a naravno želimo vidjeti je li se zadatak uspješno obavio, moramo otići u izbornik Monitor te izabrati podizbornik Tasks gdje ćemo vidjeti status našeg zadatka kao što prikazuje Slika 6.1.

Action	State	Result	▼ Started at	Ended at	User
Package Profile Update for vli001.gzaop.local	stopped	success	2018-04-29 15:01:33 ...	2018-04-29 15:04:08 ...	foreman_admin

Slika 6.1 Prikaz status obavljenog zadatka



Slika 6.2 Udaljeno upravljanje paketima

6.2. Udaljeno upravljanje zakrpama

Na sličan način kao što radimo upravljanje paketima možemo raditi upravljanje zakrpama, no to ne radimo tako da označujemo svaki poslužitelj posebno ili više poslužitelja već željene poslužitelje grupiramo u kolekcije poslužitelja kako smo to radili u poglavlju 5.10 Izrada kolekcija poslužitelja. Na taj način možemo grupirati poslužitelje po nekim zajedničkim faktorima odnosno možemo raditi upravljanje zakrpama prema životnim okolinama poslužitelja. Na dva načina možemo provjeriti trebamo li instalirati zakrpe, prvi način je da odemo u podizbornik *Content Hosts* te pogledamo stupac *Installable Update* koji nam prikazuje broj paketa kojima je potrebna zakrpa odnosno osvježavanje, vidjeti na Slika 6.3. Drugi način je da u izborniku *Hosts* odemo u podizbornik kolekcija poslužitelja (engl. Host Collections) te odaberemo željenu kolekciju i kod ponude akcija koje želimo napraviti odaberemo *Errata Installation*. Tada se otvori prozor koji pokazuje tip i opis errate te koliko i koje poslužitelje zahvaća, pogledati Slika 6.4.

Slika 6.3 Prikaz broja errata

Content Host Errata Management

Type	Id	Title	Issued	Affected Hosts
security	RHSA-2018:0483	Important: dhcp security update	2018-03-12	2
bugfix	RHBA-2018:0422	nfs-utils bug fix update	2018-03-06	2
bugfix	RHBA-2018:0417	ksh bug fix update	2018-03-06	2
bugfix	RHBA-2018:0401	libtevent bug fix update	2018-03-06	2
security	RHSA-2018:0395	Important: kernel security and bug fix update	2018-03-06	2
bugfix	RHBA-2018:0416	systemd bug fix update	2018-03-06	2
bugfix	RHBA-2018:0402	sssd bug fix update	2018-03-06	2
bugfix	RHBA-2018:0408	gcc bug fix update	2018-03-06	2

Slika 6.4 Upravljanje erratama

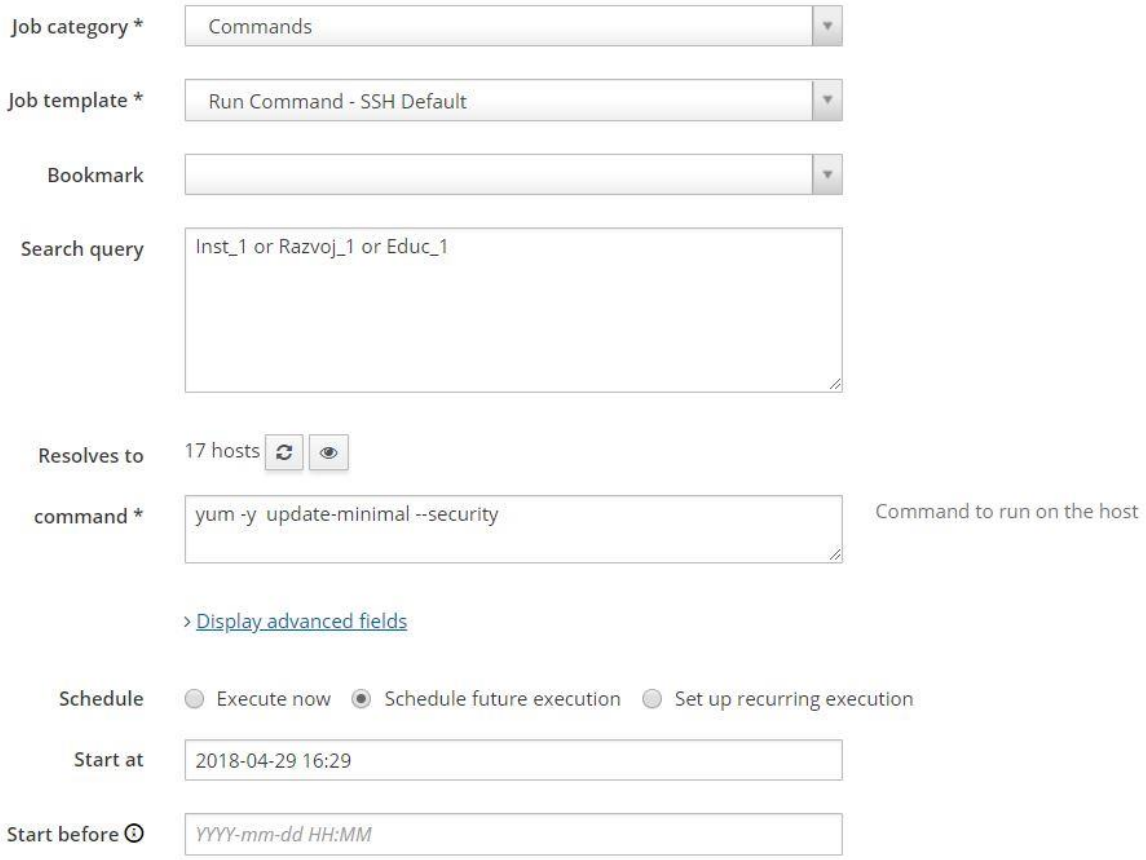
Nakon što odaberemo željene tipove errata u gornjem desnom kutu se nalazi gumb za instalaciju. Za provjeru statusa ponovo odemo u izbornik *Monitor* te podizbornik *Task*.

6.3. Zakazivanje instalacije zakrpi

Ni jedan od prijašnjih načina upravljanja zakrpama nije krivi, no zahtjeva prisutnost inženjera. Što ako želimo pustiti instalaciju zakrpi van radnog vremena? Kako bi doskočili tom poslovnom zahtjevu u pripremi smo instalirali aplikaciju za udaljeno upravljanje zvanu Remote Execution. Ta aplikacija nam omogućuje da napravimo raspored, odnosno da zakažemo koje zakrpe želimo instalirati na koje poslužitelje i kada. Vrlo je jednostavna za korištenje jer zahtjeva samo poznavanje Linux naredbi. Kako bi se izradio raspored instaliranja potrebno je otići u izbornik *Monitor* te odabrati podizbornik *Jobs* gdje ćemo u gornjem desnom kutu odabrati gumb *Run Job*. Tada nam se otvara novi prozor kao na SLICI, gdje sam ja u polje *Search query* upisao imena triju kolekcija poslužitelja koji se nalaze u tri različite životne okoline. Foreman je prepoznao o kojim se poslužiteljima radi što možemo vidjeti ako pritisnemo ikonicu oka ili broj poslužitelja koji piše u polju *Resolves to*. U polje *command* moramo upisati naredbu koju želimo primijeniti na željene poslužitelje, to mogu biti jednostavne komande poput `yum update` koji radi osvježavanje svih paketa do kompleksnijih koji će instalirati samo sigurnosne zakrpe kao što se može vidjeti u mom primjeru. Nakon što smo izabrali naredbu potrebno je postaviti datum i vrijeme puštanja zakrpi što radimo tako da izaberemo drugu opciju od tri ponuđene, odnosno *Schedule future execution*. Ono što je manje vjerojatno je da ćemo koristiti treću opciju gdje možemo ciklični rasporediti instalaciju zakrpi jer sigurno želimo imati utjecaj na izbor zakrpi. Kako bi pogledati status instalacije moramo otići u izbornik *Monitor* te odabrati podizbornik *Jobs* gdje ćemo vidjeti zadatke kao na Slika 6.5.

Description	Status	Succeeded	Failed	Pending	Total hosts	Start
Run sudo yum -y update sudo reboot	queued	—	—	—	—	in 19 days
Run sudo yum -y update sudo reboot	queued	—	—	—	—	in 18 days
Run sudo yum -y update sudo reboot	queued	—	—	—	—	in 16 days
Run sudo yum -y update sudo reboot	queued	—	—	—	—	in 15 days
Run sudo yum -y update sudo reboot	queued	—	—	—	—	in 9 days
Run sudo yum -y update sudo reboot	queued	—	—	—	—	in 8 days

Slika 6.5 Status zakazanih zadataka



Slika 6.6 Pokretanje zakazane instalacije sigurnosnih zakrpa

Zaključak

Kako se broj poslužitelja u poduzeću u kojem radim počeo svakim danom povećavati, počeli smo prelaziti na virtualne poslužitelje i tako nam je svakim danom postajalo sve jasnije da će nam trebati neki centralni sustav koji će moći pratiti službene nadogradnje, zakrpe te na kraju s kojeg ćemo moći upravljati većem brojem poslužitelja odjednom. Kako smo imali licence za Red Hat operacijski sustav, njima smo se obratili za preporuku te su nam predložili svoj sustav, Satellite. Taj sustav je zadovoljavao sve naše potrebe, no cijena licenci je bila previsoka pa smo krenuli gledati jeftinija, odnosno potpuno besplatna rješenja otvorenog koda. Tako smo pronašli Foreman, koji je besplatan program pokretan entuzijazmom zajednice te čak i velikih igrača na tržištu poput samog Red Hat-a.

Ja sam si uzeo zadatak instalirati i testirati Foreman. Kako je sam program vrlo kompleksan, sastoji se od mnogo servisa te popratnih programa, sam proces istraživanja je trajao je dosta dugo jer sam morao naučiti kako rade ti popratni programi koji zapravo grade Foreman. Kada sam završio sa istraživanjem, mislio sam da sad kreće lakši dio sa instalacijom, no ubrzo sam se razuvjerio. Kako je ovo program koji razvija zajednica, česte su promjene na svim programima, a još su češći različiti bug-ovi. Dokumentacija instalacije je također vrlo štura te objašnjava samo osnovnu instalaciju dok se za više informacija moraju konzultirati različiti forumi te se mora testirati na „vlastitoj koži“. U testnoj fazi često sam se znao vraćati unatrag jer sam shvatio da sam neke stvari u pripremi krivo posložio dok za druge nisam ni znao pa ih nisam uopće ni uključio. Ovime želim reći da je priprema Foremana zapravo ključan dio instalacije jer ako primjerice LVM-ove ne formatiramo XFS sustavom, kad-tad će se zapuniti inode-ovi te ćemo biti u problemima. Nakon same pripreme i instalacije, sučelje Foremana je vrlo smisleno što olakšava rad, no na nama je da prilagodimo dalje Foreman našoj organizaciji. Ovdje sam vrlo često koristio dokumentaciju Rad Hat-ovog Satellite programa jer sam Foreman u tom dijelu nema dokumentacije uopće. U samom radu Foreman je u 95% slučajeva vrlo stabilan, u onih 5% slučajeva ocrtava se cijena programa, odnosno činjenice da je program besplatan te samim time nemate službenu podršku za bugove na koje korištenjem naiđemo.

Za kraj bih htio zaključiti isplativost samog Foremana. Neupitna je njegova isplativost zbog velikog broja funkcionalnosti koje nudi, te zbog posla kojeg pomoću Foremana možemo napraviti, odnosno ne moramo više raditi. Foreman mogu koristiti od najmanjih pa do

najvećih organizacija u ovisnosti o tome koliko su spremni uložiti inženjerskih sati u pripremu, instalaciju te održavanje. Male organizacije si definitivno ne mogu priuštiti programe poput Satellite-a, pa se Foreman nameće kao logičan izbor. Velike organizacije već imaju izbor koji se svodi na to žele li imati službenu podršku te tako rasteretiti svoje inženjere. Moje osobno mišljenje je da je Foreman odličan program za ono za što je namijenjen te sam zadovoljan što sam imao prilike sudjelovati u njegovoj implementaciji.

Popis kratica

LVM	*Logical Volume Manager*	Upravitelj logičkih volumena
CV	*Content View*	Pregled sadržaja
CCV	Composite *Content View*	Kompozitni pregled sadržaja
IT	*Information Technology*	Informacijske tehnologije
CLI	*Command-Line Interface*	Komandno linijsko sučelje
API	*Application Program Interface*	Aplikacijsko programsko sučelje
GUI	*Graphical User Interface*	Korisničko grafičko sučelje
DNS	*Domain Name Service*	Domenski sustav imena
DHCP	*Dynamic Host Configuration Protocol*	Protokol dinamičkog konfiguriranja računala
PXE	*Preboot Execution Environment*	Okruženje za izvršavanje prije dizanja
TFTP	*Trivial File Transfer Protocol*	Jednostavan protokol za prijenos datoteka
UDP	*User Datagram Protocol*	Korisnički datagram protokol
TCP	*Transmission Control Protocol*	Protokol kontrole prijenosa
RPM	*Red-hat Package Manager*	Upravitelj paketa
ISO	*International Organization for Standardization*	Internacionalna organizacija za standardizaciju
GB	*Gigabyte*	Gigabajt
MB	*Megabyte*	Megabajt
NFS	*Network File System*	Mrežni datotečni sustav
XFS	*X File System*	XFS datotečni sustav
EXT4	*Extended File System*	EXT4 datotečni sustav
SSL	*Secure Socket Layer*	Metoda kriptiranja web prometa
CA	*Certificate Authority*	Certifikacijski autoritet
UUID	*Universally Unique Identifier*	Jedinstvena oznaka
SSH	*Secure Shell*	Sigurni mrežni protokol
LDAP	*Lightweight Directory Access Protocol*	Imenički protokol
DN	*Distinguished name*	Prepoznatljivo ime

Popis slika

Slika 2.1 Foreman Smart-Proxy arhitektura (izvor www.theforeman.org) 2
Slika 2.2 Foreman arhitektura sustava (izvor www.redhat.com) 4
Slika 2.3 Pregled sadržaja kroz životne cikluse (izvor www.redhat.com) 8
Slika 2.4 Struktura poslužiteljskih grupa (izvor www.redhat.com) 11
Slika 3.1 Prikaz LVM sheme (izvor linux.macrofinances.com) 16
Slika 3.2 Foreman logički prikaz portova (izvor www.redhat.com) 22
Slika 4.1 Uspješna instalacija Foremana 26
Slika 4.2 Status virt-who agenta 27
Slika 5.1 Početna stranica Foremana 30
Slika 5.2 Promjena lozinke i vremenske zone korisnika 31
Slika 5.3 Odabir izrađene organizacije 32
Slika 5.4 Konfiguracija LDAP poslužitelja 33
Slika 5.5 Konfiguracija korisničkog računa za pristup LDAP-u 33
Slika 5.6 Konfiguracija LDAP atributa 34
Slika 5.7 Životne okoline organizacije 35
Slika 5.8 Red Hat repozitoriji 36
Slika 5.9 Otkrivanje repozitorija 37
Slika 5.10 Broj paketa nakon sinkronizacije 38
Slika 5.11 Izrada repozitorija 39
Slika 5.12 Plan sinkronizacije repozitorija 40
Slika 5.13 Popis repozitorija 41
Slika 5.14 Lista CV-ova 42
Slika 5.15 Uspješna registracija poslužitelja u Foreman 44
Slika 6.1 Prikaz status obavljenog zadatka 45

Slika 6.2 Udaljeno upravljanje paketima 46
Slika 6.3 Prikaz broja errata 47
Slika 6.4 Upravljanje erratama 47
Slika 6.5 Status zakazanih zadataka 48
Slika 6.6 Pokretanje zakazane instalacije sigurnosnih zakrpa 49

Popis tablica

Tablica 3.1 Sistemski zahtjevi Foreman poslužitelja 13

Tablica 3.2 Količina prostora podijeljena po ključnim direktorijima 15

Tablica 3.3 Propušteni portovi na vatrozidu..... 21

Popis kôdova

Kod 3.1 Izrada logičkih volumena 17
Kod 3.2 Formatiranje logičkih volumena 18
Kod 3.3 Izrada direktorija za montiranje 18
Kod 3.4 Podešavanje konfiguracije u /etc/fstab datoteci 19
Kod 3.5 Mount direktorija 19
Kod 3.6 Povećavanje swap particije 20
Kod 3.7 Propuštanje portova kroz vatrozid 21
Kod 3.8 Konfiguracija SELinux portova 22
Kod 3.9 Konfiguracijska datoteka za SSL certifikat 23
Kod 4.1 Omogućavanje Red Hat repozitorija 24
Kod 4.2 Instalacija repozitorija 25
Kod 4.3 Instalacija potrebnih aplikacija 25
Kod 4.4 Pokretanje katello-certs-check skripte 26
Kod 4.5 Naredba za instalaciju Foremana 26
Kod 4.6 Konfiguracija Virt-Who datoteke 27
Kod 4.7 Instalacija Hammer CLI dodatka 28
Kod 4.8 Hammer CLI konfiguracijska datoteka 28
Kod 4.9 Instalacija Remote Execution dodatka 29
Kod 5.1 Izrada organizacije 32
Kod 5.2 Izrada životnih okolina organizacije 35
Kod 5.3 Izrada, publiciranje i promocija CV-a 41
Kod 5.4 Izrada kompozitnih CV-ova 42
Kod 5.5 Izrada aktivacijskog ključa 43
Kod 5.6 Izrada kolekcija poslužitelja 44

Literatura

[1] https://access.redhat.com/documentation/en-us/red_hat_satellite/6.3/pdf/architecture_guide/architecture-guide.pdf, 2018.

[2] https://access.redhat.com/documentation/en-us/red_hat_satellite/6.3/pdf/quick_start_guide/quick-start-guide.pdf, 2018.

[3] https://access.redhat.com/documentation/en-us/red_hat_satellite/6.3/pdf/installation_guide/installation-guide.pdf, 2018.

[4] https://access.redhat.com/documentation/en-us/red_hat_satellite/6.3/pdf/administering_red_hat_satellite/administering-red-hat-satellite.pdf, 2018.

[5] https://access.redhat.com/documentation/en-us/red_hat_satellite/6.3/pdf/managing_hosts/managing-hosts.pdf, 2018

[6] https://access.redhat.com/documentation/en-us/red_hat_satellite/6.3/pdf/content_management_guide/content-management-guide.pdf, 2018.

[7] https://access.redhat.com/documentation/en-us/red_hat_satellite/6.3/pdf/errata_management_guide/errata-management-guide.pdf, 2018.

[8] https://access.redhat.com/documentation/en-us/red_hat_satellite/6.3/pdf/hammer_cli_guide/hammer-cli-guide.pdf, 2018.

[9] https://access.redhat.com/articles/3362091, 2018.

[10] https://access.redhat.com/solutions/10021, 2018.

[11] https://github.com/theforeman/hammer-cli-foreman/blob/master/doc/configuration.md, 2018.

[12] https://access.redhat.com/solutions/1554783, 2018.

[13] https://theforeman.org/2013/11/hammer-cli-for-foreman-part-i-setup.html, 2018.

[14] https://access.redhat.com/documentation/en-us/red_hat_satellite/6.2/html-single/host_configuration_guide/#sect-Red_Hat_Satellite-Host_Configuration_Guide-Registration-Installing_the_Katello_Agent, 2018.

[15] https://access.redhat.com/solutions/2099421, 2018.

[16] https://access.redhat.com/documentation/en-us/red_hat_satellite/6.3/html-single/virtual_instances_guide/#configuring-virt-who, 2018.

[17] https://www.centos.org/docs/5/html/5.1/Deployment_Guide/s1-lvm-intro-whatis.html, 2018.

[18] https://theforeman.org/manuals/1.16/index.html, 2018.

[19] https://theforeman.org/plugins/katello/3.5/installation/index.html, 2018.