Implementacija sigurnosnih politika na operacijskom sustavu Linux

Novak, Marko

Undergraduate thesis / Završni rad

2023

Degree Grantor / Ustanova koja je dodijelila akademski / stručni stupanj: **University of Zagreb, Faculty of Organization and Informatics / Sveučilište u Zagrebu, Fakultet organizacije i informatike**

Permanent link / Trajna poveznica: https://urn.nsk.hr/urn:nbn:hr:211:081737

Rights / Prava: Attribution 3.0 Unported/Imenovanje 3.0

Download date / Datum preuzimanja: **2024-05-13**

SVEUČILIŠTE U ZAGREBU
FAKULTET ORGANIZACIJE I INFORMATIKE
VARAŽDIN

Repository / Repozitorij:

Faculty of Organization and Informatics - Digital Repository

zir.nsk.hr

DIGITALNI AKADEMSKI ARHIVI I REPOZITORIJI

SVEUČILIŠTE U ZAGREBU
FAKULTET ORGANIZACIJE I INFORMATIKE
VARAŽDIN

Marko Novak

IMPLEMENTACIJA SIGURNOSNIH POLITIKA NA OPERACIJSKOM SUSTAVU LINUX

ZAVRŠNI RAD

Varaždin, 2023.

SVEUČILIŠTE U ZAGREBU

FAKULTET ORGANIZACIJE I INFORMATIKE

V A R A Ž D I N

Marko Novak

JMBAG: 0016149418

Studij: Informacijski i poslovni sustavi

IMPLEMENTACIJA SIGURNOSNIH POLITIKA NA OPERACIJSKOM SUSTAVU LINUX

ZAVRŠNI RAD

Mentor:

Prof. dr. sc. Ivan Magdalenić

Varaždin, kolovoz 2023.

Marko Novak

Izjava o izvornosti

Izjavljujem da je moj završni rad izvorni rezultat mojeg rada te da se u izradi istoga nisam koristio drugim izvorima osim onima koji su u njemu navedeni. Za izradu rada su korištene etički prikladne i prihvatljive metode i tehnike rada.

Autor potvrdio prihvaćanjem odredbi u sustavu FOI-radovi

Sažetak

Ovaj završni rad donosi pregled trenutnog stanja na tržištu softvera za centralizirano upravljanje sigurnosnim politikama na operacijskom sustavu Linux, uz detaljan pregled nekih od najpopularnijih rješenja. Prvi dio završnog rada odnosi sa ne problem centraliziranog upravljanja identitetom u organizacijama. Prvo su ukratko navedeni i opisani glavni alati od kojih se sastoji većina glavnih alata koji su detaljnije opisani u nastavku. Drugi dio završnog rada odnosi se na detaljniji pregled nekih od najpopularnijih rješenja za centralizirano upravljanje konfiguracijom uz opis njihovih funkcionalnosti i posebnosti kao i neki primjeri njihovog korištenja. Zadnji dio rada je pogled na jedno od dostupnih rješenja u oblaku.

Ključne riječi: Linux; Active Directory; Group Policy; Configuration Management; Identity Management; Sigurnosne politike; FreeIPA; JumpCloud

Sadržaj

1. Uvod 1

2. Centralizirano upravljanje identitetom 2
2.1. Pojedini alati 2
2.1.1. Lightweight Directory Access Protocol (LDAP) 2
2.1.2. System Security Services Daemon (SSSD) 3
2.1.3. Samba 3
2.2. Red Hat IdM / FreeIPA 4
2.2.1. Korisnici 4
2.2.2. Domaćini (računala) 5
2.2.3. Netgroups 5
2.2.4. ID pogledi 5
2.2.5. Automatski član (Automember) 5
2.2.6. Podređeni ID-jevi 6
2.2.7. Servisi 6
2.2.8. Sudo pravila 6
2.2.8.1. Koji korisnici 7
2.2.8.2. Koja računala 7
2.2.8.3. Koje naredbe 7
2.2.8.4. Kao tko 8
2.2.8.5. Opcije 9
2.2.8.6. Redoslijed 9
2.2.9. Upravljanje pristupom na temelju domaćina (HBAC) 9
2.2.9.1. HBAC pravila 9
2.2.9.2. Testiranje 11
2.2.10. SELinux korisničke mape 11
2.2.11. Politike lozinka (Password Policies) 12
2.2.12. Upravljanje Kerberos kartama 13
2.2.13. Upravljanje pristupom na temelju uloga (RBAC) 13
2.2.14. Ostalo 13
2.3. Univention Corporate Server 13
2.3.1. Korisnici, računala i grupe 14
2.3.2. Upravljanje softverom servera 15
2.3.3. Univention Configuration Registry 16
2.3.4. Politike lozinka (Password Policies) 17
2.3.5. Napredne i nepodržane mogućnosti 17

2.3.6. Ostalo 18

3. Centralizirano upravljanje konfiguracijom 19
3.1. Usporedba alata 19
3.2. Ansible 20
3.2.1. Pojmovnik 20
3.2.2. Način rada 20
3.2.3. Primjeri 21
3.2.3.1. Inventar 21
3.2.3.2. Playbook 22
3.2.3.3. Malo naprednije mogućnosti 22
3.2.4. Access Control List (ACL) 24
3.2.5. Integracija s Red Hat IdM 25
3.2.6. ansible-pull 25
3.2.7. AWX (komercijalno Ansible Automation Controller) 26
3.2.8. Ostalo 26
3.3. SaltStack 27
3.3.1. Pojmovnik 27
3.3.2. Spajanje SaltStack Miniona na Master 27
3.3.3. Osnovni primjeri 28
3.3.4. Malo naprednije mogućnosti 29
3.3.5. Uyuni (SUSE Manager) 30
3.3.5.1. Upravljanje ključevima 30
3.3.5.2. Slanje naredbi i raspored 30

4. Directory-as-a-Service (DaaS) 32
4.1. JumpCloud 32
4.1.1. Dodavanje računala 32
4.1.2. Korisnici 32
4.1.3. Lozinke 33
4.1.4. Grupe 33
4.1.5. Upravljanje politikama 34
4.1.6. Ostalo 35

5. Zaključak 36

Popis literature 39

Popis slika 41

Popis popis tablica 42

1. Uvod

Kad je neka organizacija još mlada i ima malen broj računala, upravljanje informatičkom infrastrukturom je relativno jednostavno. U većini slučajeva je dovoljno da zadužena osoba tretira svako računalo kao zasebni entitet i sve što je potrebno uskladiti među računalima radi ručno za pojedino računalo. Sve postavke se ručno podešavaju za pojedino računalo, a korisnički računi su ponekad jedinstveni za svako računalo. U malim organizacijama to ne predstavlja veliki problem jer nije previše kompleksno za održavanje, ali kada organizacija naraste, sama količina računala i informacija, kao i novi zahtjevi i poslovne politike, drastično povećavaju kompleksnost cijele informatičke strukture.

Nije teško uskladiti konfiguraciju malog broja računala, a ni naknadne promjene konfiguracije ne zahtijevaju previše vremena, ali kada se govori o stotinama ili čak tisućama računalima, podešavati svako od njih ručno postaje bolan i mukotrpan posao za cijeli tim ljudi. Problemu ne pomaže to što svaka osoba može doći u neku neočekivanu situaciju gdje mora privremeno promijeniti zadane postavke, nakon čega ih zaboravi vratiti u početno stanje.

Velik broj ljudi koji imaju jednak pristup svim računalima je često značajan sigurnosni rizik. Taj rizik se ublažuje na način da se detaljnije podese dozvole pristupa za svaku osobu. To se najčešće radi zasebnim korisničkim računima koje je potrebno dodati i podesiti na svakom računalu koje ta osoba smije koristiti. Naravno da tu nije kraj. Osoba može promijeniti ulogu unutar organizacije, a time i dozvole pristupa, što zahtjeva ponovno podešavanje svih korisničkih računa na svim računalima. Ponovne promjene računa također mogu prouzročiti i sigurnosne politike organizacije, kao što je redovita promjena lozinki i brisanje korisničkih računa u slučaju da osoba napusti organizaciju.

Kad se sve to uzme u obzir i pomnoži se nekoliko stotina ili tisuća računala s nekoliko stotina ili tisuća korisnika, očito je da takav način rada postaje dugoročno neodrživ. Za rješenje tog problema su kreirani razni alati koji omogućuju centralizirano upravljanje korisničkim računima i postavkama samih računala. Naravno da su najpopularniji alati, Microsoft Active Directory i Group Policy, namijenjeni najpopularnijem operacijskom sustavu, Microsoft Windows. Pošto Microsoft Windows ima najviše korisnika, upravljanje korisnicima i računalima na kojima se nalazi Windows je dobro podržano, detaljno testirano i kroz godine usavršeno, ali to isto se ne može reći i za operacijski sustav Linux, čiji je udio na tržištu znatno manji i koji ima mnogo različitih distribucija koje zasebno održavaju i unaprjeđuju dobar dio svog sustava.

Microsoft Active Directory ima relativno slabu podršku za Linux, a Group Policy gotovo nikakvu, barem u odnosu na Windows. Kada daleko najpopularniji alat ne nudi željene mogućnosti, potrebno je pronaći alternative. Na sreću, iako se Linux distribucije razlikuju u mnogo stvari, njihovi stvoritelji i održavatelji su svjesni značajnosti inteoperabilnosti između sustava i centraliziranog upravljanja sustavima. Iz tih se razloga većina distribucija drži dogovorenih standarda za inteoperabilnost, što je u našem slučaju odlična vijest, jer nam omogućuje jednostavno centralizirano upravljanje računalima s raznim Linux distribucijama.

2. Centralizirano upravljanje identitetom

Upravljanje identitetom je (u ovom kontekstu) kombinacija autentikacije i autorizacije. Prvo se provjerava je li korisnik onaj koji tvrdi da jest, a nakon toga ima li taj korisnik prava za odraditi određenu akciju.

Neko osnovno centralizirano upravljanje autentikacijom može se implementirati na način da se potrebni korisnički identifikator (korisnička imena, E-mail adrese i slično) i pripadajuće lozinke spremaju u zajedničkoj bazi podataka iz koje se svaki put čitaju podaci kada je to potrebno. U malim organizacijama to nije problem, ali kada se drastično poveća broj korisnika, a time i korisničkih računa i količine upita na bazu, taj jednostavan model postaje neefikasan. Za rješenje tog problema postoje posebno dizajnirane i optimizirane baze podataka koje omogućuju replikaciju kako bi se smanjio napor za pojedinu bazu i korisnički alati koji privremeno lokalno pohranjuju određene podatke dohvaćene iz baze.

Centralizirano upravljanje autorizacijom bi se također moglo na najjednostavniji način implementirati tako što bi se, zajedno s korisničkim računima, u bazu spremale dozvole vezane uz te korisničke račune. Takav način autorizacije brzo postaje neefikasan, nepregledan i neodrživ, iz tog se razloga skupine dozvola često grupiraju prema nekom kriteriju kao što je uloga osobe unutar organizacije, uloga računala, geografska lokacija osobe ili računala, način pristupa računalima i još mnogi drugi.

Za sve te probleme postoje pojedini alati koji ih rješavaju ili ublažuju, kao i alati koji omogućuju jednostavno i usklađeno upravljanje tim pojedinim alatima.

2.1. Pojedini alati

Ovdje su navedeni neki od najpopularnijih alata za rješenje prethodno navedenih problema. Oni se često koriste zajedno i do neke mjere su inteoperabilni.

2.1.1. Lightweight Directory Access Protocol (LDAP)

Popularnost relacijskih baza podataka je opće poznata. Ako ste se ikad susreli s ikojom bazom podataka, vjerojatno je bila relacijska i koristila SQL za čitanje i manipulaciju podataka. Takve baze su dovoljno dobre za spremanje većine vrsta strukturiranih podataka, ali naš slučaj je malo specifičan jer zahtjeva veoma brzu i čestu interakciju s korisnicima. Zamislite da svaki put kada korisnici žele promijeniti tekući direktorij (otvoriti novu mapu), otvoriti neki dokument ili pokrenuti neki alat, moraju čekati par sekundi da se potvrdi imaju li dovoljna prava pristupa. To bi značajno usporavalo rad korisnika i vjerojatno nekima išlo na živce.

Pošto se korisnički podaci ne mijenjaju toliko često, dobar izbor za njihovu pohranu su direktoriji, ili točnije, baze podataka koje spremaju podatke slično kao direktoriji. Takve su baze, za razliku od relacijskih, najviše optimizirane za brzo pronalaženje i čitanje podataka, a manje za brzo dodavanje i izmjenu podataka. Najpopularniji standard za takve baze je Lightweight

Directory Access Protocol (LDAP) i postoje razne implementacije, ali najpopularnije su OpenLDAP i 389 Directory Server.

2.1.2. System Security Services Daemon (SSSD)

Kako se ne bi svaka aplikacija ručno morala brinuti za komunikaciju s LDAP bazom, postoji alat System Security Services Daemon (SSSD) koji se brine za to. SSSD radi nekoliko stvari, ali najbitnije su integracija s Pluggable Authentication Moduleom (PAM) i Name Service Switchem (NSS), te privremena lokalna pohrana podataka

Na Linux sustavima, većina programa podržava modul zvan Pluggable Authentication Module (PAM), koji se brine za autentikaciju korisnika i dobiveni rezultat proslijedi aplikaciji [1]. To omogućuje korištenje različitih metoda autentikacije (otisak prsta, tokeni, certifikati, itd.) bez da ih aplikacija direktno implementira. PAM kao skup biblioteka je dobro integriran s LDAP i Kerberos servisima, kao i `sudo` naredbom, što developerima i korisnicima aplikacija znatno olakšava posao, jer developeri ne moraju ručno implementirati sve poželjne način autentikacije, a korisnici dobivaju jedno centralizirano mjesto na kojem mogu jednostavno prilagođavati načine autentikacije za razne aplikacije. Vrijedi napomenuti i da PAM ima korisnu mogućnost stvaranja "home" direktorija za svakog novo prijavljenog korisnika [2].

Moglo bi se reći da, što je PAM za autentikaciju, to je NSS za autorizaciju. Name Service Switch (NSS) služi za centralnu konfiguraciju repozitorija iz kojeg sustav i aplikacije vuku potrebne podatke o korisnicima [3]. Na primjer, kada želite dodati nešto u dokument, sustav provjerava imate li prava za to (npr. pripadate li grupi kojoj je dopuštena izmjena dokumenta). Tako NSS omogućuje specifikaciju odabrane LDAP baze podataka za traženje potrebnih informacija.

Kombinacija rješenja za kako se prijaviti i gdje pogledati potrebne podatke te njihovo centralizirano upravljanje je već dovoljan razlog za korištenje SSSD-a, ali SSSD još nudi i privremenu lokalnu pohranu određenih podataka u lokalni "cache". Kada se korisnik prvi put prijavi u sustav, SSSD provjerava postoji li veza do, u našem slučaju, LDAP baze. U slučaju da postoji, tamo se traže korisnički podaci, a u slučaju da, na primjer, nestane internetska veza, a LDAP baza se nalazi na udaljenom računalu, SSSD neko vrijeme sprema korisnike koji su se prije uspješno prijavili na računalo u svoj lokalni cache. Tako se korisnici mogu prijaviti i u slučaju nestanka veze, problema s poslužiteljem ili samom LDAP bazom. Također se i nakon svakog odgovora podaci o korisniku kao što su grupe kojima pripada spremaju u cache i neko vrijeme primarno iz njega čitaju kako bi se smanjio napor nad bazom [4].

2.1.3. Samba

Najjednostavnije rečeno, Samba je skup alata za inteoperabilnost između Windows i Linux računala. Od verzije 4.0, Samba može poslužiti kao Active Directory Domain Controller (AD DC) na razini Windows Server 2008 R2, što je više nego dovoljno za upravljanje sofisticiranim poduzećima koja koriste Windows 10/11 [5]. Drugim riječima, Samba omogućuje korištenje velikog dijela funkcionalnost Microsoft Active Directroya, ali bez potrebe za Windows Serverom.

Sastoji se od dva glavna programa: smbd i nmbd, koji se brinu za upravljanje datotekama i printerima, autentikaciju i autorizaciju, razdjeljivanje imena (engl. name resolution) i najavu servisa (za popis servisa svih računala) [6].

Mali je problem što je Samba server potrebno ručno konfigurirati. Za manje programe to nije problem, ali Samba definitivno ne spada u tu kategoriju. Potrebno je jako puno ručnog uređivanja raznih datoteka, što znači brojne mogućnosti za ljudske greške. Na sreću je dovoljno popularan alat da ima vrlo dobru integraciju s ostalim alatima iz ove domene.

2.2. Red Hat IdM / FreeIPA

Red Hat Identity Management (IdM) se prije zvao IPA, što je dobro vidljivo na CLI naredbama koje se koriste. Upstream projekt od IdM-a je FreeIPA [7]. Gotovo su identični, samo je IdM stabilnija verzija za koju Red Hat pruža komercijalnu podršku, dok FreeIPA nudi najnovije funkcionalnosti koje nisu uvijek nužno stabilne.

Red Hat Identity Management (IdM) pruža centraliziran i ujedinjen način za upravljanje spremištima identiteta, autentikacijom, politikama i politikama autorizacije u domenama baziranim na Linux-u [8]. Sastoji se od nekoliko različitih alata, uključujući [9]:

- 389 Directory Server - implementacija LDAP servera
- MIT Kerberos - implementacija Kerberos protokola za mrežnu autentikaciju
- NTP (Network Time Protocol) - protokol za sinkronizaciju vremena računala; veoma bitno za Kerberos certifikate
- DNS - BIND server
- Dogtag sustav za certifikate - lokalni Ceritficate Authority
- SSSD - daemon za centralizirano upravljanje identitetom, autentikacijom i autorizacijom [10]

Svi ti alati zajedno stvaraju jedan iznimno moćan i sveobuhvatan alat koji omogućuje upravljanje velikim brojem različitih računala i integraciju s Microsoft Active Directory. Iako je i Ubuntu podržan kao Operacijski sustav na kojem se može instalirati IdM server, preporučeno je koristiti Red Hat distribucije (RHEL, Fedora, CentOS, itd.). Dolje su navedene uglavnom mogućnosti za direktno upravljanje sigurnosnim politikama. Valja napomenuti da se sve radnje mogu izvršiti i automatizirati putem naredbene linije (Command-line Interface, CLI).

2.2.1. Korisnici

IdM omogućuje, između ostalog, kreiranja, mijenjanje, onemogućavanje i brisanje korisnika. Jedini obavezni podaci za svakog korisnika su ime i prezime, a uz to još postoje mnogi drugi osobni, poslovni i tehnički podaci koji se mogu nadodati. Moguće je i unaprijed kreirati korisnike, npr. nove zaposlenike prije nego što im je potreban korisnički račun. Ovisno o situaciji,

možda postoji šansa da će se obrisani korisnik vratiti. Za takvu situaciju postoje "očuvani" korisnici koji su gotovo identični obrisanim korisnicima, samo što ih je moguće jednostavno vratiti u prave korisnike.

2.2.2. Domaćini (računala)

Moguće je definirati tehničke (npr. na koji način se korisnici smiju prijaviti na računalo) i poslovne (npr. lokacija) podatke za pojedino računalo. Neki od podataka su platforma i operacijski sustav, SSH javni ključevi, MAC adresa, tko smije čitati i kreirati Kerberos keytab tog računala i je li to računalo vjerodostojno za delegaciju nekih poslova od IdM-a.

2.2.3. Netgroups

Radi olakšanja posla, IdM omogućuje grupiranje korisnika, računala i mnogih drugih stvari, a Netgroups su posebna vrsta grupa. To je koncept od, sada zastarjelog, Network Information Service (NIS) protokola koji se prije zvao Yellow Pages (YP). Netgroups pod LDAP protokolom služe za inteoperabilnost, ali i stvaranje grupa korisnika i računala, pretežito u svrhe pogodnosti.

2.2.4. ID pogledi

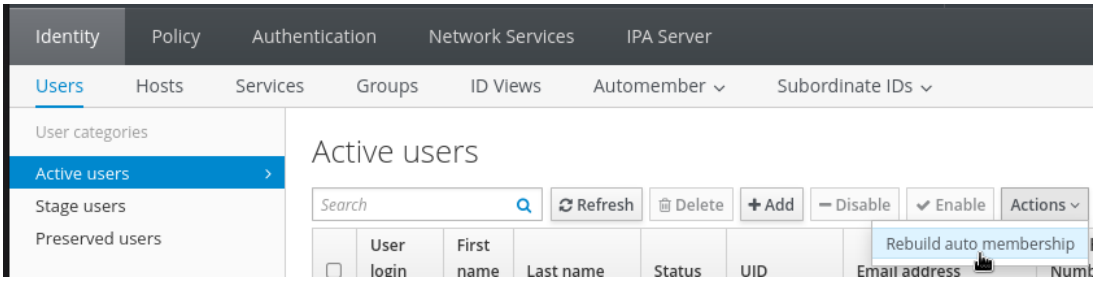
Kod korisnika se može definirati mnogo stvari, kao na primjer njihov "home" direktorij, ali što ako taj direktorij iz bilo kojeg razloga nije prikladan za sva računala? Tome slože ID pogledi. Velik dio postavki vezanih uz korisnički račun su vezani uz korisnikov korisnički identifikator (UID) i grupni identifikator (GID). ID pogledima možemo promijeniti te, a time i ostale vrijednosti za određene korisnike na određenim računalima. Postoji još mnogo situacija u kojima bi to bilo korisno, kao na primjer kod uvođenja sustava za upravljanje identitetom gdje su korisnici prije toga bili ručno kreirani na svakom računalu i često se njihovi korisnički identifikatori ne podudaraju. To je malo naprednija mogućnost, ali je korisno znati da postoji jer je u određenim situacijama od neprocjenjive vrijednosti.

2.2.5. Automatski član (Automember)

IdM omogućuje definiranje pravila prema kojima se korisnici ili računala automatski dodaju u određene grupe. U grupe se dodaje ako odabrani atribut iz baze odgovara zadanom RegEx izrazu. Na slici ispod je jednostavna grupa u koju će biti dodani svi korisnici s titulama koje počinju na veliko slovo "C" i imaju ukupno 3 do 7 velikih slova, osim CEO-a.

Slika 1: Jednostavan Automember primjer [autorski rad]

Svi novi korisnici za koje to pravilo vrijedi će automatski biti dodani u odabranu grupu, ali već postojeći korisnici neće biti automatski dodani. Za to je potrebno ići pod "Identity" -> "Users" i gore desno pod "Actions" odabrati "Rebuild auto membership". Tada će i postojeći korisnici biti dodani u prikladne grupe. Isto to vrijedi i za računala samo umjesto pod "Users" treba ići pod "Hosts".



Slika 2: Osvježavanje automatskih članstva korisnika [autorski rad]

2.2.6. Podređeni ID-jevi

Podređeni ID-jevi (engl. Subordinate IDs) su korisnički ili grupni identifikatori (UID, GID) koji su podređeni nekom korisniku. To se uglavnom koristi za upravljanje kontejnerima bez potreba za administratorskim pravima[11].

2.2.7. Servisi

Postavke za pojedine servise su veoma slične kao postavke pojedinih domaćina (računala), samo bez podataka koje se ne odnose na njih (npr. verzija operacijskog sustava). Glavna dodatna mogućnosti je opcija za dopuštanje servisu da autenticira korisnike.

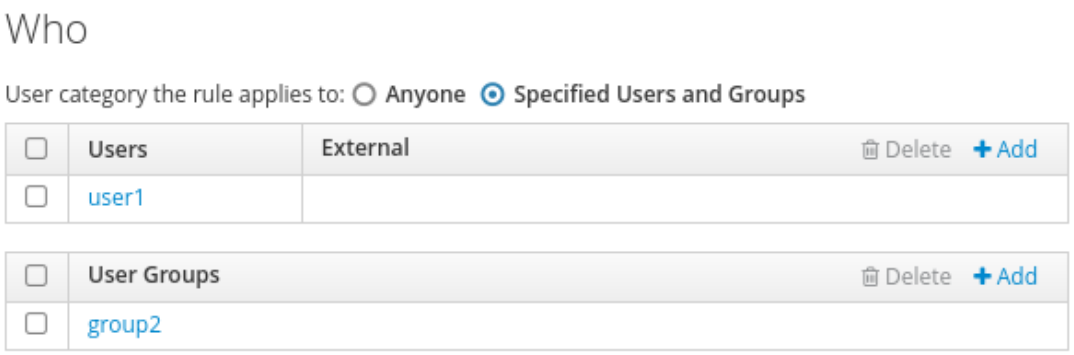
2.2.8. Sudo pravila

Naredba `sudo` ima dobru integraciju sa SSSD koja omogućuje čitanje dozvola iz LDAP baze. Tako je moguće da IdM upravlja dozvolama za izvršavanje naredba za koje su potrebna administratorska (ili neka druga) prava korištenjem popularnog alata `sudo`. Moguće je podesiti na razini korisnika i grupa točno tko, koje naredbe smije izvršavati. `sudo` ne služi isključivo

za pokretanje naredbi kao administrator, već se može koristiti za pokretanje naredbi kao bilo koji korisnik. Dozvole za to tko može koristiti `sudo` kao koji drugi korisnik se također mogu jednostavno podesiti koristeći IdM.

2.2.8.1. Koji korisnici

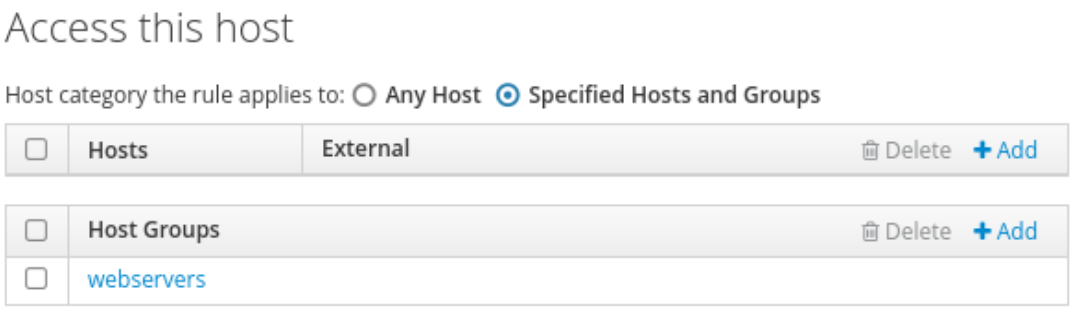
Moguće je definirati na koje korisnike i grupe se točno odnosi definirano pravilo, to jest, koji korisnici (ne)smiju na navedeni način koristiti navedene naredbe.



Slika 3: Na koje korisnike se odnosi ovo sudo pravilo [autorski rad]

2.2.8.2. Koja računala

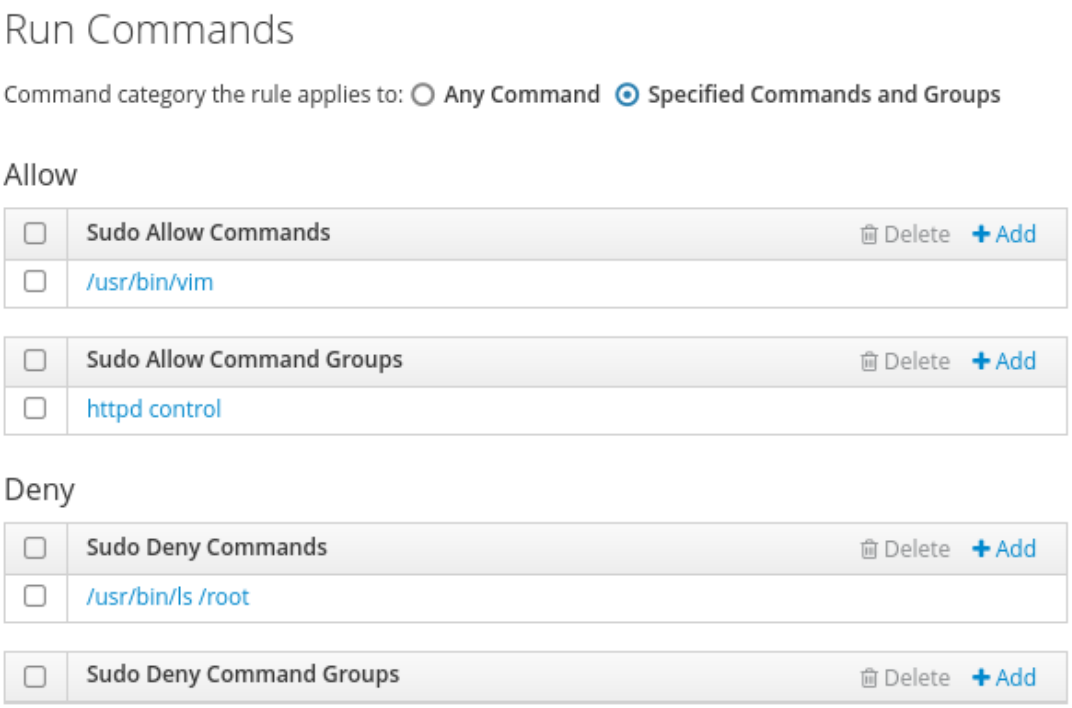
Moguće je definirati na koje računalo (domaćina) ili grupe računala se odnosi definirano pravilo, to jest, na kojim računalima navedeni korisnici smiju na navedeni način koristiti navedene naredbe.



Slika 4: Na koja računala se odnosi ovo sudo pravilo [autorski rad]

2.2.8.3. Koje naredbe

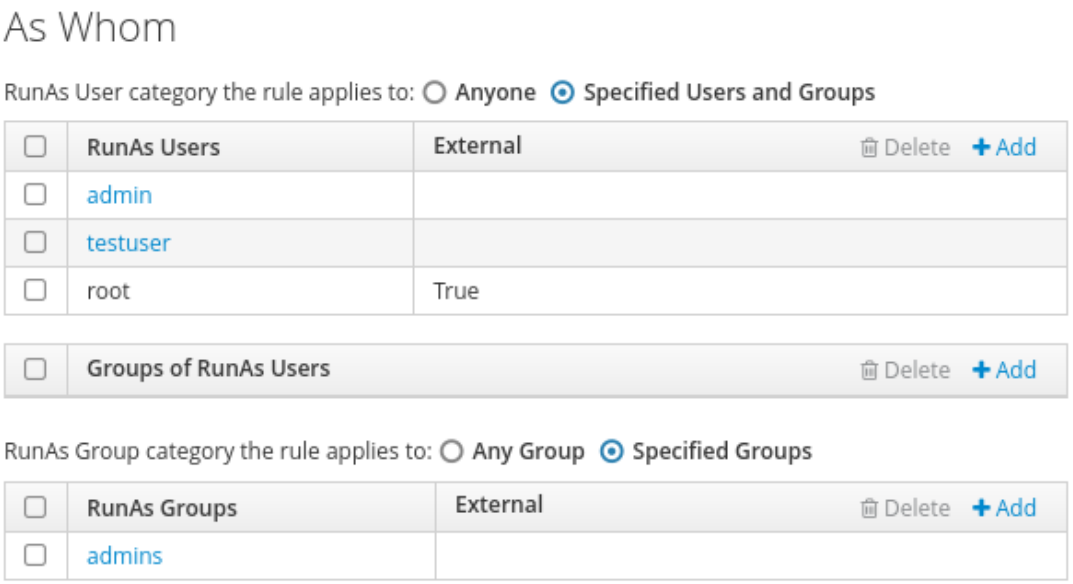
Moguće je definirati točne naredbe ili grupe naredbi koje su dozvoljene i zabranjene.



Slika 5: Na koja naredbe se odnosi ovo sudo pravilo [autorski rad]

2.2.8.4. Kao tko

`sudo` ne služi isključivo za pokretanje naredbi s administratorskim pravima, već služi za pokretanje naredbi kao neki drugi korisnik. Moguće je definirati koji korisnici i grupe korisnika su dozvoljeni za to.



Slika 6: U ime kojih korisnika se smiju koristiti navedene naredbe [autorski rad]

2.2.8.5. Opcije

Moguće je navesti dodatne opcije kod pokretanja `sudo` naredbe koje su podržane u `sudoers` programu. Ispod je navedena opcija `!authenticate` koja označava da nije potrebno upisati korisničku lozinku kod korištenja naredbi u ovom pravilu.

Slika 7: Dodatne opcije za pokretanje sudo naredbi [autorski rad]

2.2.8.6. Redoslijed

U slučaju konfliktnih pravila, moguće je, kao i u standardnoj `sudoers` datoteci, definirati redoslijed pravila. U `sudoers` datoteci, poštuje se zadnje definirano pravilo, a ovdje se poštuje pravilo s najvećim brojem. Pojednostavljeno: ako pravilo 1 za nešto kaže "DA", a pravilo 2 kaže "NE", odgovor je "NE".

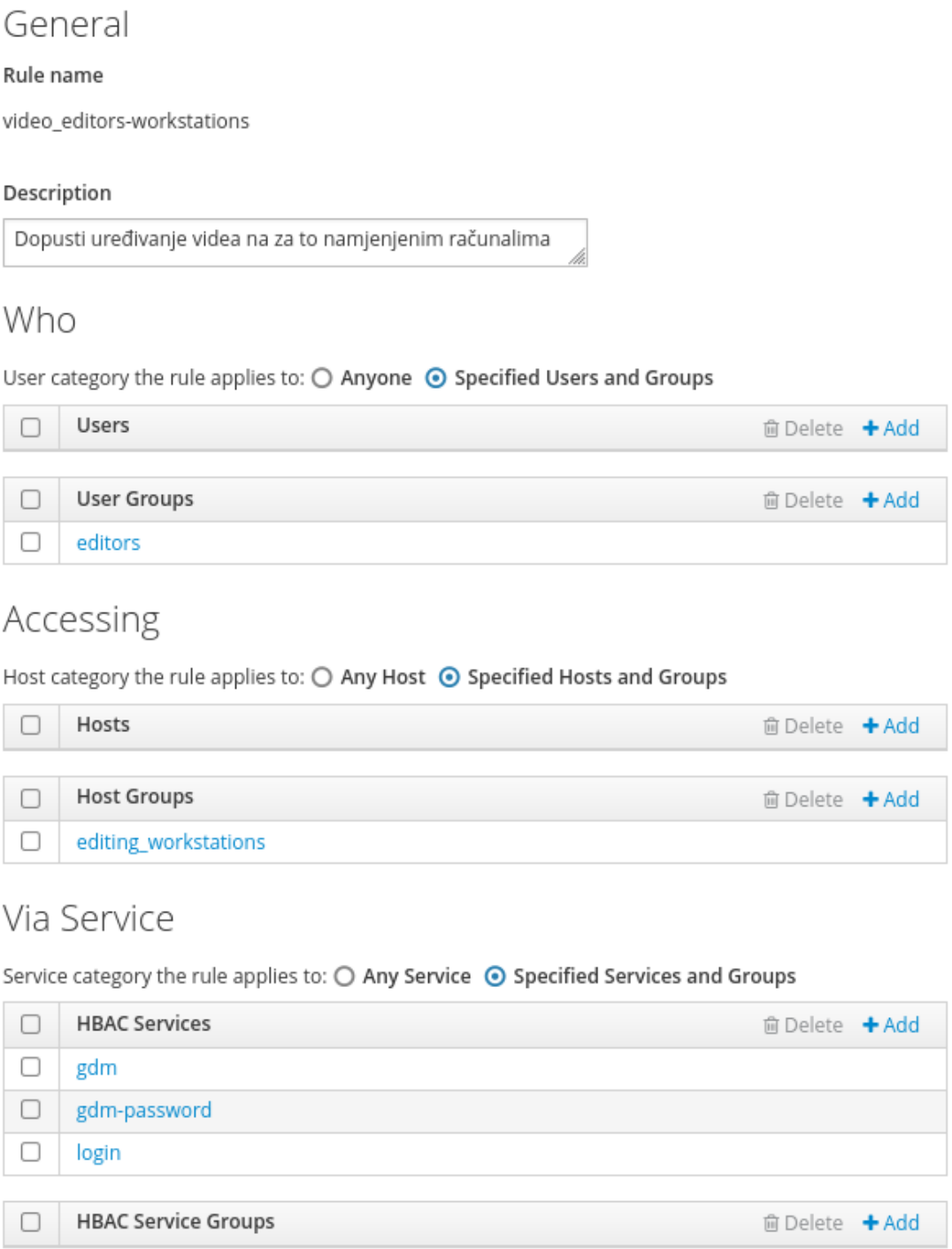
Slika 8: Definiranje pozicije u redu izvršavanja [autorski rad]

2.2.9. Upravljanje pristupom na temelju domaćina (HBAC)

Host-Based Access Control određuje tko se i na koji način može spojiti na koje računalo (domaćina). HBAC radi na principu "least-privilege", što u ovom kontekstu znači da korisnik nema pristup računalu ako nije eksplicitno navedeno. Iz tog razloga odmah nakon instalacije postoji HBAC pravilo `allow_all`, koje dopušta svim korisnicima da na sve načina pristupe svim računalima. Nakon uspješnog definiranja svojih novih pravila, potrebno je obrisati to `allow_all` pravilo kako bi sve pravilno funkcioniralo.

2.2.9.1. HBAC pravila

U ovom jednostavnom primjeru se uređivačima videa dopušta pristup računalima koja su za to namijenjena putem obične prijave.



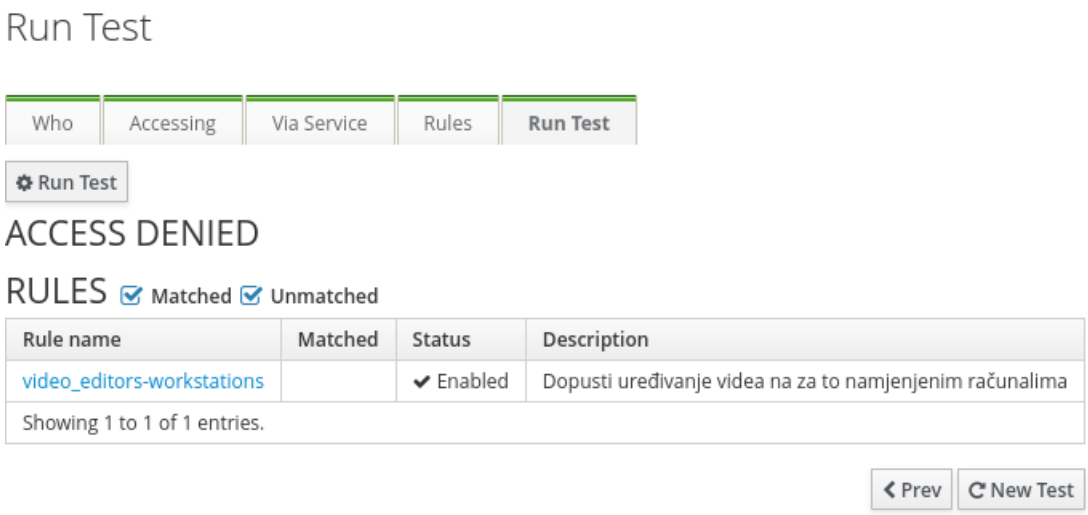
Slika 9: Jednostavan HBAC primjer [autorski rad]

Servisi `gdm` i `gdm-password` su PAM servisi koje koristi Gnome Display Manager. To je jedan od programa koji se pokreće kada se uključe navedena računala i kroz njega se korisnici prijavljuju u svoje korisničke račune. IdM ima unaprijed definirane najčešće PAM servise, ali se mogu dodati i vlastiti.

2.2.9.2. Testiranje

IdM ima jednostavan alat za testiranje HBAC pravila. Moguće je odabrati korisnika, računalo, servis i pravila te pokrenuti test koji provjerava može li se odabrani korisnik na odabrani način prijaviti na odabrano računalo slijedeći odabrana pravila.

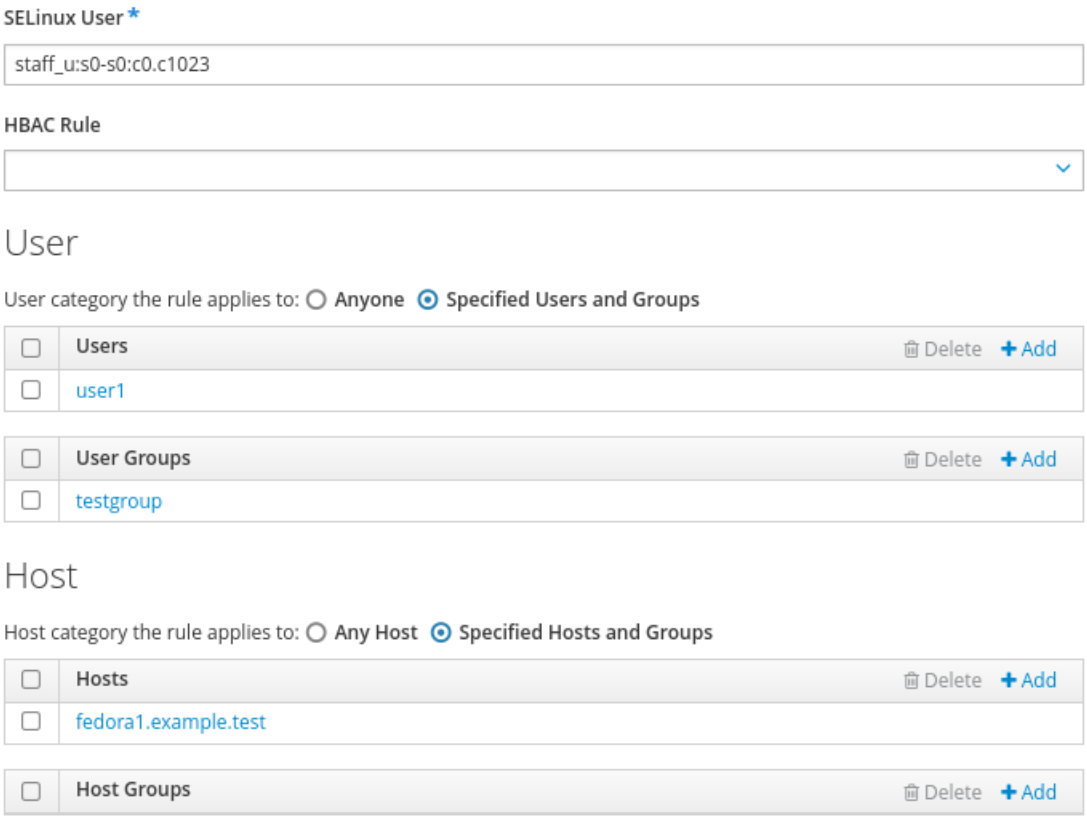
Ovo je jednostavan primjer gdje se korisnik koji nije u grupi `editors` pokušava prijaviti na jedno računalo putem običnog logina uz uključeno prethodno stvoreno HBAC pravilo. Pristup mu naravno nije odobren.



Slika 10: Jednostavan HBAC Test [autorski rad]

2.2.10. SELinux korisničke mape

IdM ima dobru integraciju sa Kernel modulom Security Enhanced Linux (SELinux) koji daje specifičan i detaljan način za upravljanje kontrolom pristupa na računalu. Moguće je definirati jedno HBAC pravilo na koje da se odnosi ili ručno navesti korisnike i računala. SELinux može veoma brzo postati veoma kompleksan, iz tog je razloga potrebno prvo detaljno proučiti upute i napraviti nekoliko lokalnih primjera prije nego li se krenu stvarati putem IdM-a. Ispod je prikazan jednostavan primjer gdje su posebno navedeni korisnici i računala (nije moguće istovremeno navesti HBAC pravilo i korisnike/računala).



Slika 11: Jednostavan SELinux primjer [autorski rad]

2.2.11. Politike lozinka (Password Policies)

IdM ima unaprijed definiranu globalnu politiku lozinka koja se odnosi na sve korisnike. Nju je moguće mijenjati, a moguće je i dodati vlastite politike za odabrane grupe korisnika (tada se globalna politika ignorira za korisnike u toj grupi). Moguće je definirati:

- Maksimalno trajanje lozinke (nakon toga je potrebno obnoviti lozinku)
- Minimalno trajanje između izmjena lozinki
- Broj lozinki koji se sprema u povijest korištenih lozinki (korisnik ne može ponovo iskoristiti lozinke iz svoje povijesti)
- Vrste različitih karaktera koje lozinka mora sadržavati
  - mogući unosi su brojevi od 0 do 5 koji označavaju broj potrebnih različitih karaktera
  - vrste karaktera su:
    - velika slova
    - mala slova
    - brojevi
    - znakovi (,.* itd.)
    - ostali UTF-8 karakteri

- Minimalnu duljinu
- Maksimalan broj neuspješnih pokušaja prijave prije nego što se korisnički račun zaključa
- Interval poništavanja neuspješnih pokušaja prijave
- Trajanje blokiranja (nakon maksimalnog broja neuspješnih pokušaja)
- Prioritet (za korisnike koji su u više grupa)
- Limit za "milosrdnu" prijavu (broj dozvoljenih prijava s nekom lozinkom nakon što joj je isteklo maksimalno trajanje)

2.2.12. Upravljanje Kerberos kartama

Ukratko, Kerberos je protokol kojim se prijenos lozinki preko mreže svodi na minimum. Umjesto lozinki se prenose certifikati koji potvrđuju da netko ima pravo pristupa nekom servisu. IdM omogućuje postavljanje trajanja životnog vijeka certifikata i vremena do kojeg se certifikati mnogu obnoviti. Te postavke se također mogu specificirati ovisno o tome kakav način autentikacije je korišten za dobivanje certifikata (npr. ako je korištena obična lozinka ili dvofaktorska autentikacija).

2.2.13. Upravljanje pristupom na temelju uloga (RBAC)

Skupine dozvola koje se grupiraju prema ulogama osoba unutar organizacije prikladno se nazivaju "upravljanje pristupom na temelju uloga" (engl. Role-Based Access Control), skraćeno RBAC. To se odnosi na pisanje. čitanje, brisanje i mijenjanje podataka unutar LDAP baze.

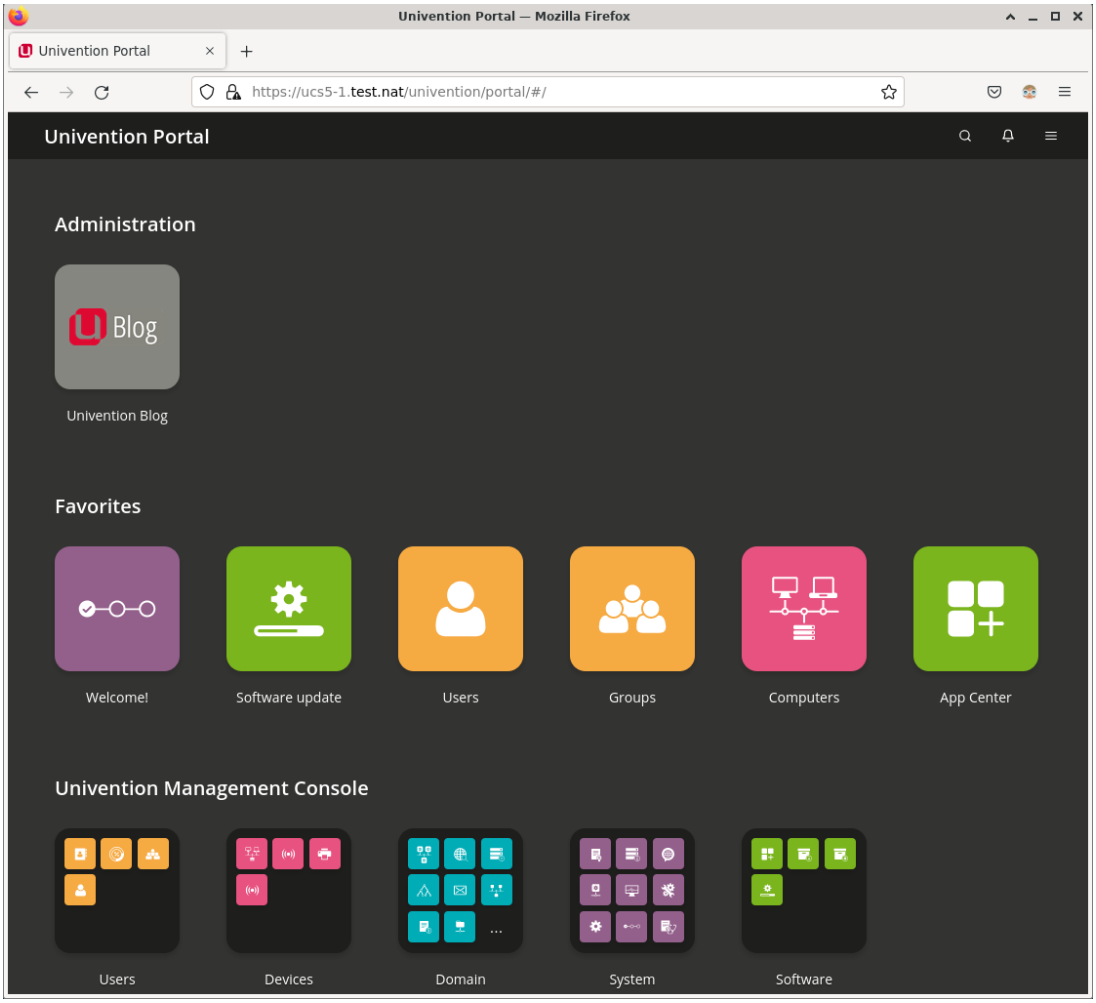
2.2.14. Ostalo

Red Hat IdM / FreeIPA podržava još hrpu opcija kao što su upravljanje DNS serverima i zonama, upravljanje i "povjerenje" prema domenama, pregled trenutne topologije, detaljna konfiguracija samog servera, komunikacija putem API-ja i još mnoge druge opcije različitih razina složenosti i korisnosti. Ovdje su navedene samo one koje se najviše bave sigurnosnim politikama. Više informacija i detaljnu dokumentaciju možete pronaći na Red Hat-ovim stranicama na poveznici `https://access.redhat.com/products/identity-management/`.

2.3. Univention Corporate Server

Izabran je iz razloga što za Sambu postoji samo komercijalna podrška od treće strane te samo postavljanje i upravljanje Samba serverom zahtjeva mnogo ručnog rada i znanja. Postoji nekoliko "all-in-one" rješenja koja pokušavaju što bolje emulirati jednostavnost Microsoftovog Active Directorya, ali Univention Corporate Server (UCS) je jedan od najjednostavnijih i već se koristi kod malih, srednjih i velikih organizacija [12].

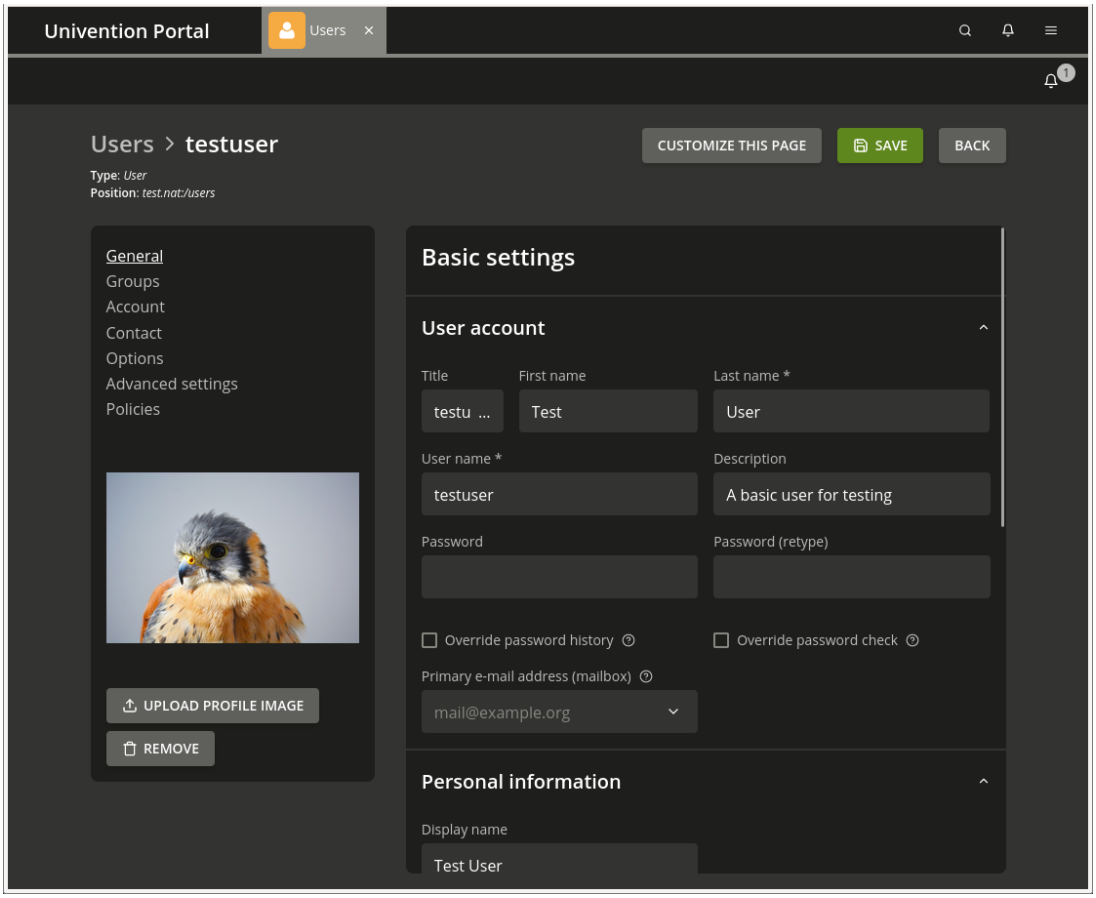
UCS je cijeli operacijski sustav baziran na Debian-u koji je potrebno instalirati direktno na računalo ili kao virtualni stroj. Iako je potrebno instalirati cijeli novi operacijski sustav, rijetko ga je potrebno koristiti kao takvog. Većina radnji se obavlja preko web preglednika putem Univention Portala koristeći Univention Management Console (UMC), koja pruža jako uredno i jednostavno grafičko sučelje za upravljanje svim najčešće korištenim mogućnostima. Univention Portalu se pristupa putem IP adrese ili naziva UCS servera (u slučaju da je pravilno konfiguriran DNS server za lokalno računalo).



Slika 12: Univention Management Console servera ucs5-1 na mreži test.nat [autorski rad]

2.3.1. Korisnici, računala i grupe

Moguć je jednostavan pregled svih korisnika kao i njihovo jednostavno te napredno pretraživanje. Klikom na dodavanje korisnika ili promjenom postojećeg korisnika otvara se kartica s detaljima o korisniku. Uz osnovne podatke o korisniku, moguće je i dodati profilnu sliku, razne osobne podatke, poslovne podatke i grupe kojima korisnik pripada. Također je moguće vidjeti i mijenjati informacije o tome je li račun deaktiviran, kada mu ističe valjanost, je li zaključan zbog previše netočnih unosa lozinke, je li potrebna promjena lozinke kod sljedeće prijave, je li i kada potvrđena aktivacija računa, koji je korisnikov "home" direktorij i preferirana ljuska, identifikator korisnika (UID) i identifikator grupe (GID), i još mnogo toga.



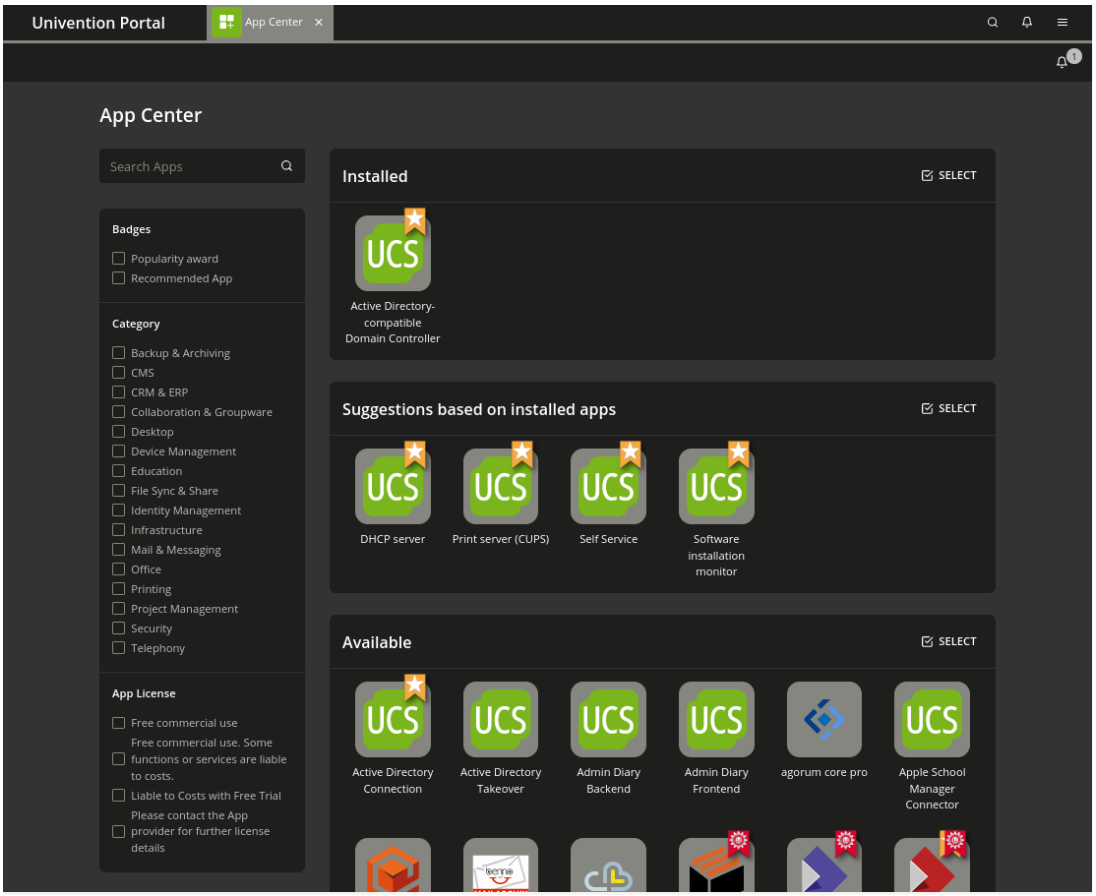
Slika 13: Promjena korisničkih podataka [autorski rad]

Podacima o računalima se upravlja na sličan način kao i kod korisnika. Glavna razlika je vrsta podataka. Tako se za računala mogu podesiti osnovni podaci (naziv, operacijski sustav, IP adresa, itd.) i napredni podaci (DNS Forward i Reverse Lookup Zone, DNS alias, DHCP, itd.).

Slično kao i kod samih korisnika i računala, za grupe postoje neke osnovne postavke (naziv, članovi, itd.) i naprednije postavke (npr. da član te grupe automatski postane član nekih drugih grupa). Valja napomenuti da članovi grupa mogu biti korisnici i računala.

2.3.2. Upravljanje softverom servera

App Center je mjesto na kojem je moguće preuzeti dodatne "aplikacije" koje proširuju funkcionalnosti servera (npr. Element, GitLab, Jitsi Meet, Keycloak, Nextcloud Hub, OnlyOffice Docs, ownCloud, Rocket.Chat, itd.). Neke aplikacije su besplatne, ali za neke je potrebno imati prihvatljivu UCS licencu (koju je potrebno kupiti kako biste, među ostalim, dobili podršku za softver).

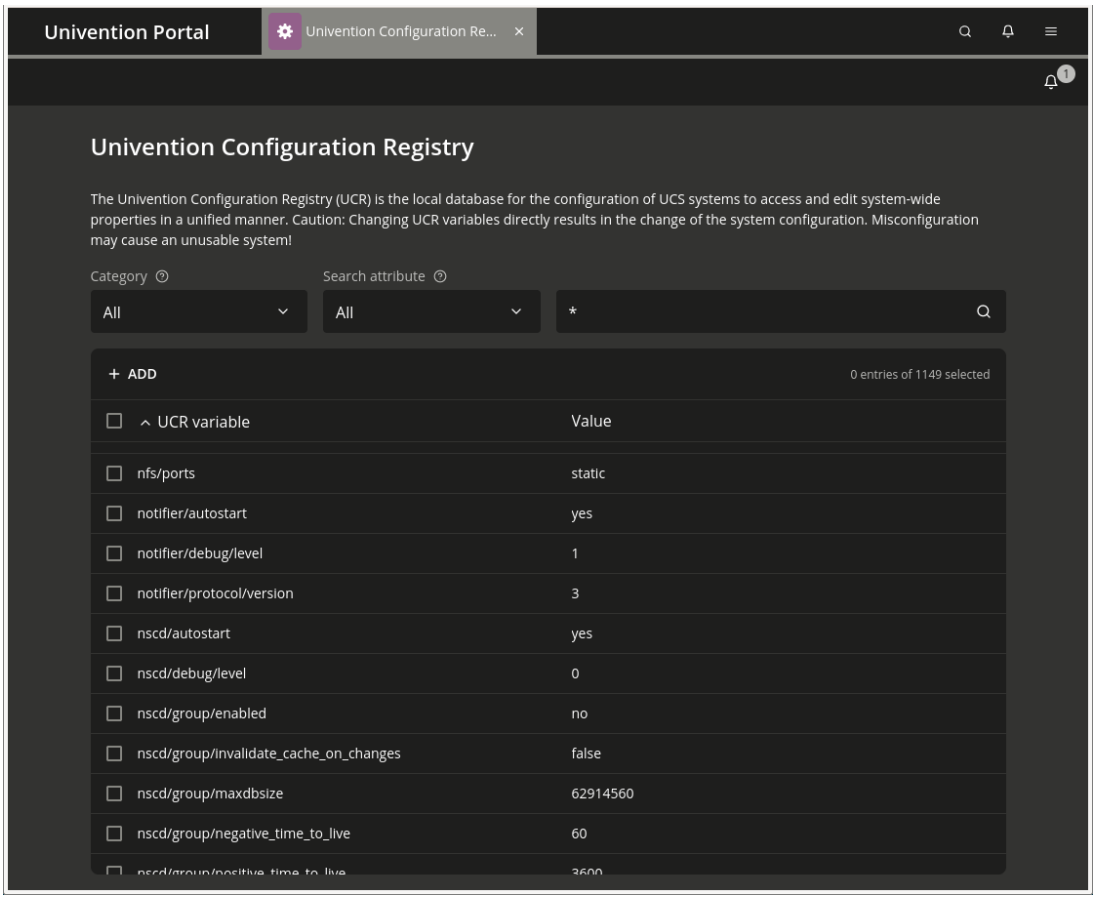


Slika 14: App Center [autorski rad]

Osim ponuđenih "aplikacija", UCS omogućuje pregledan način za upravljanje paketima i repozitorijima. Upravljanje paketima omogućuje jednostavan pregled svih dostupnih paketa s osnovnim i detaljnim opisom i pretraživanje prema kategorijama, nazivu i opisu. Upravljanje repozitorijima omogućuje upravljanje postojećim i dodavanje novih repozitorija na jednostavan način (moglo bi se reći da to je GUI za rad s `/etc/apt/sources.list` i `sources.list.d`).

2.3.3. Univention Configuration Registry

Velik fokus UCS-a je na Windows računalima i inteoperabilnosti servisa između Windows i Linux računala, ali u ovom je radu fokus na Linux. Taj fokus na Windows kao zamjena za Active Directory Domain Controller je dobro vidljiv kod upravljanja postavkama samog UCS servera. Za podešavanje postavki se koristi Univention Configuration Registry (UCR) koji ima veliku sličnost s Registry-em na Windows računalima i služi za podešavanje gotov svih postavki servera za koje ne postoji zasebno grafičko sučelje (Kerberos, ssh, Firewall, itd.).



Slika 15: Univention Configuration Registry [autorski rad]

2.3.4. Politike lozinka (Password Policies)

Osnovno upravljanje politikama lozinka je veoma jednostavno. Na Univention Management Console pod "Domain" treba izabrati "Policies" i u gornjem lijevom kutu tablice kliknuti "+ ADD". Tada se otvori prozor gdje se odabere tip politike (u ovom slučaju "Policy: Passwords") i LDAP kontejner po potrebi. Nakon unosa tih informacija potrebno je upisati naziv politike i neke od osnovnih parametra koji su: duljina lozinke, rok do isteka lozinke u danima i duljina povijesti (lozinke u povijesti se ne mogu ponovo iskoristiti). Za kraj je još moguće odlučiti hoće li se provjeravati kvaliteta lozinke. Postavke procjene kvalitete lozinke su identične za sve lozinke na serveru. Iz tog se razloga te postavke podešavaju u Univention Configuration Registryu (UCR). Najlakše ih je pronaći tražeći kategoriju "Password settings" i pretraživanjem riječi "quality". Postavke s "credit" označuju minimalan broj takve vrste znakova u lozinki. Postavka "forbidden/chars" označuje koje karaktere je zabranjeno koristiti, a "required/chars" koje je nužno koristiti. Detaljan opis svake postavke može se dobiti prelaskom miša preko nje ili klikom na nju.

2.3.5. Napredne i nepodržane mogućnosti

UCS nudi grafičko sučelje za velik dio funkcionalnosti, ali je za neke stvari potrebno koristiti ne ili djelomično podržana rješenje (npr. paketi iz cool-solutions repozitorija za upravljanje

`sudo` politikama) ili za to namijenjene alate. Tako Samba pruža neke mogućnosti vezane uz sigurnosne politike za Linux koje je potrebno podešavati ručno ili koristeći alat `samba-tool` (npr. politike za Firefox i Chromium/Chrome kao što su početna stranica i ostale postavke).

2.3.6. Ostalo

Kako bi se računalo pridružilo domeni UCS servera, za Ubuntu postoji skripta koja automatski sve podešava i pridružuje računalo domeni. Ta skripta uz instalaciju potrebnih paketa radi i za Debian. Kod ostalih Linux distribucija potrebno je ručno povezivanje kao na običnu Samba/AD domenu (najlakše uz alat `realmd`).

Uz navedene mogućnosti za upravljanje sigurnosnim politikama, UCS također nudi i razne ugrađene servise kao npr. DNS, DHCP i E-mail te razne mogućnosti za jednostavan pregled trenutnog stanja računala preko mreže (pokrenuti procesi, postavke mreže, svojstva hardvera, jezik, itd.). Za velike organizacije postoje i mogućnosti jednostavne replikacije servera, stvaranja "backup" servera i zasebnih servera koji su zaduženi samo za upravljanje printerima i datotekama preko mreže.

3. Centralizirano upravljanje konfiguracijom

Active Directory Group Policy ima iznimno slabu podršku za Linux operacijski sustav. Niti Group Policy kao pojam ne postoji baš za Linux. Iz tog se razloga koriste alati za upravljanje konfiguracijom koji efektivno mogu raditi istu stvar i potencijalno još mnogo više.

3.1. Usporedba alata

DSL = Domain Specific Language - to je jedinstven jezik koji se koristi, u ovom slučaju, isključivo za programiranje kod navedenog alata.

Tablica 1: Usporedba alata za upravljanje konfiguracijom [autorski rad]

	Ansible	Puppet	Chef Infra	SaltStack
Napisan u jeziku:	Python [13]	Ruby [14]	Ruby [15]	Python [16]
Korišten jezik:	YAML [17]	Puppet DSL [18]	Ruby i Chef DSL [19]	YAML ili JSON [20]
Jezik predložaka:	Jinja2 [21]	Embedded Puppet i/ili Embedded Ruby [22]	Embedded Ruby [23]	Jinja2 ili Mako ili Wempy [20]
*Pretežito deklarativan / imperativan:	Deklarativan [24]	Deklarativan [18]	Imperativan [25]	Deklarativan [26]
*Pretežito potreban klijentski softver:	Ne [27]	Da [28]	Da [29]	Da [30]
*Pretežito push/pull:	Push [13]	Pull [31]	Pull [32]	Push [33]
Podržan master (server) OS:	Red Hat, Debian, Ubuntu, macOS, BSD-evi, Windows uz WSL i gotovo svi UNIX-like sustavi [34]	RHEL, CentOS, Oracle Linux, Scientific Linux, SLES, Ubuntu [35]	Amazon Linux 2, CentOS, Oracle Enterprise Linux, Red Hat Enterprise Linux, SUSE Linux Eterprise Server, Ubuntu [36]	Linux (AlmaLinux, Amazon Linux, CentOS, Debian, Fedora, Oracle Linux, Photon OS, RHEL, OpenSUSE, SLES, Ubuntu), macOS, Windows [37]
Podržan OS za klijenta (agenta):	Gotovo sve što podržava SSH [34]	Linux (RHEL i slični, SLES, Debian, Ubuntu, Amazon Linux), AIX, Solaris, Windows, macOS [35]	Linux (gotovo sve distribucije), IBM AIX, FreeBSD, NetBSD, OpenBSD, macOS, Oracle Solaris, Windows [36]	Linux (RHEL i slični, Arch Linux, Debian, FreeBSD, openSUSE, Photon OS, SLES, Ubuntu) [37]

*svi navedeni alati do neke mjere podržavaju obje opcije, ali se uglavnom koriste na jedan od tih načina

3.2. Ansible

Ansible je radikalno jednostavna platforma za IT automatizaciju koja olakšava postavljanje i održavanje sustava [38]. Iako ima neke mogućnosti upravljanja infrastrukturom, ovdje će biti isključivo riječ o upravljanju konfiguracijom kao zamjena za Active Directory Group Policy.

3.2.1. Pojmovnik

Ansible koristi neke specifične pojmove. Ovdje su navedeno samo osnovni pojmovi koji će se koristiti u ovom radu, detalji se mogu pronaći u dokumentaciji na poveznici `https://docs.ansible.com/ansible/latest/reference_appendices/glossary.html`.

- Kontrolni čvor - računalo koje kontrolira ostala računala (upravljane čvorove)
- Upravljani čvor (engl. Managed node) - računala koja kontrolira kontrolni čvor
- Inventar - popis svih upravljanih čvorova (računala) te opcionalno neke informacije o njima i neke varijable
- Playbook - datoteka koja sadrži skup Playeva
- Play - povezuje zadatke i upravljane čvorove (računala); prilagođava zadatak podacima od pojedinog računala
- Uloge - grupacije datoteka, upravljača, zadataka i ostalih stvari u logičku cjelinu (npr. uloga webserveri)
- Zadaci - pojedini zadaci unutar jednog Playa koji će se izvršiti pokretanjem
- Upravljači (engl. Handlers) - posebna vrsta zadataka koja se izvršava samo kada ih pozovu neki drugi zadaci
- Moduli - kod ili binarne datoteke koje Ansible pokreće na određenom upravljanom čvoru (računalu) kada je to potrebno
- Plugin - dodaci na Ansible koji omogućuju nove načine spajanja na upravljane čvorove, manipulaciju podacima i datotekama i još mnogo toga
- Kolekcije - grupacije modula

3.2.2. Način rada

Ansible uglavnom radi na način da se pokreću Playbookovi (bilo to ručno ili automatski) na temelju kojih se, putem SSH, ansible prijavi kao odgovarajući korisnik u svako računalo i provjerava što je potrebno učiniti. Pošto se Ansible zadaci uglavnom pišu na deklarativan način, Ansible prvo provjerava stanje računala i tek kada utvrđeno stanje ne odgovara navedenom stanju, Ansible automatski radi sve što je potrebno kako bi uskladio trenutno i navedeno stanje računala i na kraju vraća sveobuhvatan izvještaj.

3.2.3. Primjeri

3.2.3.1. Inventar

Najjednostavniji način definiranja inventara jest iz `/etc/ansible/hosts` datoteke. Tamo Ansible i automatski traži inventar ako nije drukčije navedeno. Lokaciju te datoteke je moguće trajno promijeniti u Ansible konfiguraciji ili za svaki poziv Playbooka zasebno uz `-i` opciju. Inventar može biti YAML ili INI datoteka. Dolje je jednostavan primjer koji definira jedno zasebno računalo (upravljani čvorovi) i dvije grupe računala (webservers i dbservers).

INI:

```
racunalo123.example.test

[webservers]
fedora1.example.test
192.168.123.123

[workstations]
fedora1.example.test
fedora2.example.test
```

YAML:

```
ungrouped:
  hosts:
    racunalo123.example.test:
webservers:
  hosts:
    fedora1.example.test:
    192.168.123.123:
workstations:
  hosts:
    fedora1.example.test:
    fedora2.example.test:
```

Uočite da je jedno računalo definirano svojom IPv4 adresom te da se `fedora1.example.test` istovremeno nalazi u dvije grupe.

U ovom jednostavnom primjeru, gotovo je svejedno želite li koristiti INI ili YAML sintaksu, ali kada stvari postanu malo kompleksnije, na primjer s definiranjem varijabli, preporuka je koristiti YAML radi preglednosti i mogućih komplikacija s INI formatom [39].

Gotovo svi dodatni podaci se mogu definirati koristeći varijable, ali u slučaju da, na primjer, želite da je iz sigurnosnih razloga prije svakog pokretanja Playbook-a potrebno unijeti lozinku za SSH, prilikom pokretanja možete dodati opciju `-ask-pass`, koja će zatražiti unos lozinke. Nikad nemojte spremati lozinke u čitljivom formatu! Za spremanje tajnih podataka preporuka je koristiti Ansible Vault koji dolazi uz instalaciju Ansiblea. Radi jednostavnosti i sigurnosti, uglavnom se koriste SSH ključevi koji se mogu prenijeti na bilo koji način (ručno, naredbom `ssh-copy-id`, automatski prilikom postavljanja računala, koristeći Ansible, itd.).

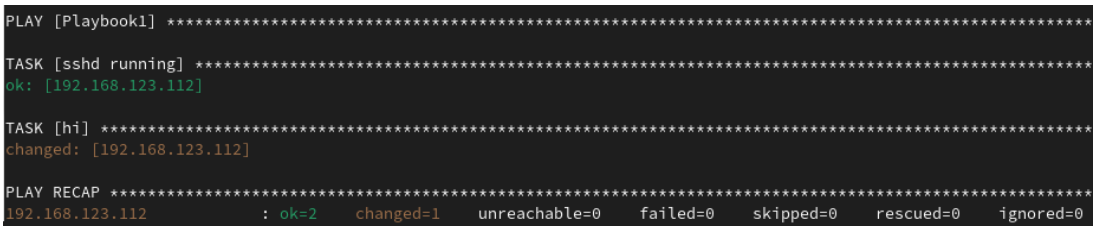
3.2.3.2. Playbook

Slijedi jednostavan primjer Playbooka koji provjerava imaju li svi webserveri pokrenut SSH daemon (naravno da imaju ako se uspješno spojimo) i postoji li `hi.txt` datoteka u "home" direktoriju. Postavlja se pitanje, u čijem "home" direktoriju? Odgovor je u "home" direktoriju od korisnika s korisničkim imenom "username". `become_user:` definira da ansible nakon spajanja na računalo želi postati taj korisnik (kao da pokrene naredbu `su username`). u slučaju da se definira samo `become: true`, Ansible pokušava postati root korisnik. U oba slučaja, ako već nije definirana negdje, moguće je prilikom poziva Playbooka dodati opciju `-K` koja daje mogućnost za unos potrebne lozinke.

```
- name: Playbook1
  hosts: webservers
  become: yes
  become_user: username
  gather_facts: no
  tasks:
  - name: sshd running
    service:
      name: sshd
      state: started
  - name: hi
    ansible.builtin.template:
      src: hi.txt
      dest: ~hi.txt
```

U slučaju da se nazivi modula ne podudaraju, moguće je navesti samo naziv (u primjeru: `service`), ali je također moguće navesti i puni naziv modula (u primjeru: `ansible.builtin.template`). `gather_facts` određuje hoće li ansible prikupljati neke korisne podatke o računalu na kojem se pokreće. U ovom jednostavnom primjeru to nije potrebno.

Kada prvi put pokrenemo gore navedeni Playbook, rezultat bi trebao izgledati ovako (naravno uz drukčiju IP adresu ovisno o postavkama mreže):



Slika 16: Rezultat prvog pokretanja jednostavnog Playbooka [autorski rad]

3.2.3.3. Malo naprednije mogućnosti

Ansible također nudi mogućnosti direktnog izvršavanja naredbi u ljusci. Za tu funkcionalnost, kao i za puno njih, postoji velik broj opcija koje možete detaljnije proučiti u dokumentaciji na poveznici `https://docs.ansible.com/ansible/latest/collections/ansible/builtin/shell_module.html`.

Kada se Playbook pokreće na računalu i opcija `gather_facts` je uključena, Ansible prikuplja veoma velik skup podataka o upravljanom čvoru (računalu) na kojem se trenutno izvršava. Ti podaci se mogu koristiti unutar Playbooka. Više informacija o tome koji se podaci prikupljaju možete pronaći na sljedećoj poveznici: `https://docs.ansible.com/ansible/latest/playbook_guide/playbooks_vars_facts.html`.

U slučaju da se dogodi neka greška, Ansible prekida izvođenje ostatka Playbooka i vraća opis dobivene greške. U slučaju da očekujemo grešku na određenom zadatku i želimo da Ansible unatoč grešci nastavi s izvršavanjem zadataka, u ciljani zadatak trebamo dodati opciju `ignore_errors: true`.

Upravljači (engl. Handlers) se izvršavaju ako su eksplicitno pozvani od zadatka i to tek nakon završetka svih zadataka. Upravljači se ne izvršavaju ako dođe do bilo kakve pogreške tijekom izvršavanja bilo kojeg zadatka, neovisno o tome je li definirano `ignore_errors: true`.

Ovaj primjer bi trebao obrisati sve datoteke u korisnikovom "Documents" direktoriju i kreirati novu datoteku `hi.txt` zatim, u slučaju da je operacijski sustav na računalu na kojem se izvršava Playbook "Fedora", u tu datoteku prvo upisati "Hello, Fedora!" te na kraju bi upravljač na dno datoteke dodao "Hi!". Naravno da je moguće postići ciljani rezultat na mnogo bolji način koristeći neke od mnoštva mogućnosti koje Ansible nudi, ali isto kao i u programiranju, često postoji mnogo načina za postići isti cilj i ovo je primjer u edukacijske svrhe.

```
- name: Playbook2
  hosts: webservers
  become: yes
  become_user: username
  gather_facts: yes
  tasks:
    - name: Empty Documents
      ansible.builtin.shell: rm ~/Documents/*
      ignore_errors: true
      notify: Insert Greeting

    - name: Greet Fedora
      shell: echo "Hello, Fedora!" >> ~/Documents/hi.txt
      when: ansible_facts['distribution'] == "Fedora

  handlers:
    - name: Insert Greeting
      shell:
        cmd: echo "Hi!" >> ~/Documents/hi.txt
```

Prije svega, pretpostavimo da je prije pokretanja direktorij "Documents" prazan i da se Playbook pokreće na računalo s Linux distribucijom "Feodra".

Prije samog pokretanja, vidljivo je da se naredbe u ljusci mogu pokretati na nekoliko različitih načina, ali sve rade istu stvar. Kada prvi put pokrenemo ovaj Playbook, zadatak "Empty Documents" će izbaciti grešku koja će se ignorirati te će se pokrenuti zadatak "Greet Fedora" koji upisuje "Hello, Fedora!" na dno datoteke `hi.txt`. Pošto ta datoteka ne postoji, automatski

će se kreirati i krajnji rezultat je da korisnik "username" na odabranom računalu u direktoriju "Documents" ima datoteku `hi.txt` u kojoj piše "Hello, Fedora!" (jer se upravljač nije izvršio zbog greške). Ansible rezultat ovog Playbooka opisuje ovako:

```
ok=3    changed=2    unreachable=0    failed=0    skipped=0    rescued=0    ignored=1
```

Slika 17: Opis rezultata nakon prvog izvršavanja Playbooka [autorski rad]

Ako još jednom pokrenemo isti playbook, rezultat i način na koji smo došli do njega će biti malo drukčiji. Prvo će zadatak "Empty Documents" obrisati sve datoteke (uključujući `hi.txt`) i pošto direktorij ovaj put nije prazan prije izvršavanja naredbe, neće baciti grešku. Nakon toga se, identično kao i prvi put, izvršava "Greet Fedora" i upisuje "Hello, Fedora!" na dno novokreirane datoteke `hi.txt`, a pošto nije došlo do greške, na samom kraju se još izvršava i upravljač "Insert Greeting" koji će na dno datoteke `hi.txt` dodati tekst "Hi!". Tako je rezultat ovog i svih daljnjih izvršavanja datoteka `hi.txt` u kojoj u prvom redu piše "Hello, Fedora!" te u drugom redu "Hi!". Ansible rezultat ovog pokretanja Playbooka opisuje ovako:

```
ok=4    changed=3    unreachable=0    failed=0    skipped=0    rescued=0    ignored=0
```

Slika 18: Opis rezultata nakon drugog izvršavanja Playbooka [autorski rad]

Ovdje je dobro vidljivo da je Ansible namijenjen pretežito imperativnom načinu rada, jer se svaki uspješno izvršeni zadatak koji koristi ljusku označuje kao "changed".

3.2.4. Access Control List (ACL)

Iako IdM/FreeIPA pruža mogućnost pravila pristupa bazirano na ulogama, Access Control Liste na razini operacijskog sustava nije moguće na jednostavan način definirati. Za to je preporuka koristiti Ansible koji dolazi s `ansible.posix.acl` modulom za takve zadatke. Access Control Liste uz malo nepažnje mogu postati kompleksni, ali postoji ograničen broj mehanizama za njihovu kontrolu. Mi ćemo se zadržat na jednostavnim primjerima, ali navedene mogućnosti su dovoljne i za upravljanje najkompleksnijim Access Control List postavkama.

```
- name: Playbook3
  hosts: webservers
  become: yes
  become_user: username
  gather_facts: no
  tasks:
  - name: Grant user1 read access to usernames hi.txt
    ansible.posix.acl:
      path: ~/Documents/hi.txt
      entity: user1
      etype: user
      permissions: r
      state: present

  - name: Set default ACL for group1 for usernames Documents directory
    ansible.posix.acl:
```

```
    path: ~/Documents
    entity: group1
    etype: group
    permissions: -wx
    default: true
    state: present

- name: Set default ACL for group2 for usernames Document directory
  ansible.posix.acl:
    path: ~/Documents
    entry: default:group:group2:rw-
    state: present

- name: Get ACL for usernames hi.txt
  ansible.posix.acl:
    path: ~/Documents/hi.txt
  register: documents_acl

- name: Output ACL
  debug:
  msg: "ACL for hi.txt: {{ documents_acl.acl }}"
```

Zahvaljujući YAML sintaksi, veoma je očito koje ACL postavke se postavljaju. Na kraju se šalje mala povratna informacija o stanju ACLa na datoteci `hi.txt`. Valja napomenuti da samo prvi zadatak mijenja ACL za `hi.txt` i to postavlja dozvole za korisnika "user1". Poruka je jednaka izlazu komande `getfacl /home/username/Documents/hi.txt`, samo u drugom formatu. Dobivena poruka je: `"msg": "ACL for hi.txt: ['user::rw-', 'user:user1:r--', 'group::r--', 'mask::r--', 'other::r--']"`. Vidimo da je definirano da "user1" ima samo dozvole čitanja.

3.2.5. Integracija s Red Hat IdM

Prethodno navedeni primjeri su tek vrh sante leda mogućnosti koje Ansible nudi. Ansible također pruža veoma dobru integraciju s velikim brojem servisa, ali onaj koji nas trenutno najviše zanima je Red Hat IdM/FreeIPA. Potrebno je instalirati paket koji ima jednak naziv neovisno radi li se o IdM ili FreeIPA. Taj paket je `ansible-freeipa` i on dodaje nove Ansible module za gotovo svaku mogućnost koju nudi IdM/FreeIPA. Ima toliko modula i Playbookova da ih se ne isplati sve nabrajati; moglo bi se reći da ima sve što bi ikad moglo zatrebati. Cijeli popis modula je dostupan na poveznici: `https://github.com/freeipa/ansible-freeipa`.

3.2.6. ansible-pull

Iako je primarno namijenjen za "pushanje" konfiguracije na računala, Ansible također podržava i mogućnost za "pullanje" Playbookova iz nekog centraliziranog repozitorija. To ima nekoliko prednosti poput mogućnosti jednostavnog verzioniranja svih Playbookova u velikim organizacijama kao i gotovo beskonačnu skalabilnost jer je samo potreban jedan centralni repozitorij iz kojeg se povlače Playbookovi, ali naravno da je moguće koristiti više sinkroniziranih

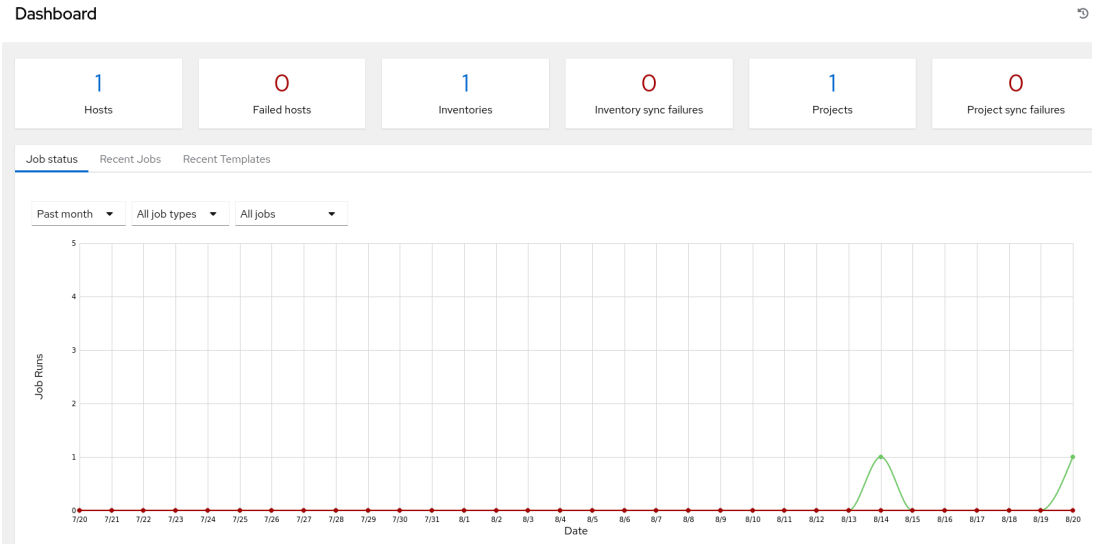
servera s usklađenim kopijama Playbookova.

Kako bi se korektno izvodio, potrebno je na ciljanom računalu (klijentu) stvoriti cronjob koji će periodički pokretati `ansible-pull` komandu s prikladnim parametrima. Mana u tome je što nije nemoguće da se dođe u situaciju gdje zadaci iz nekih Playbookova koji se izvršavaju koriste naredbe koje nisu namijenjena da se izvode paralelno, radi čega se preporuča koristiti neki vanjski raspoređivač (engl. scheduler) i/ili zaključavanje kako ne bi došlo do sukoba nekih operacija [40].

3.2.7. AWX (komercijalno Ansible Automation Controller)

Prije se komercijalna verzija zvala Ansible Tower, a sad je poznata pod imenom Ansible Automation Controller i jedan je od mnogo dijelova Ansible Automation Platforme (AAP). Slično kao i IdM/FeeIPA, AWX je besplatna verzija otvorenog koda koja ima sve nove, još ne pretjerano testirane, mogućnosti, dok Red Hat nudi podršku za Ansible Controller (u sklopu AAP-a), koji je stabilnija verzija AWX-a sa samo dobro testiranim opcijama.

AWX je ukratko GUI za Ansible, barem mu je to glavna svrha. Također nudi mogućnost postavljanja vremenom izvršavanja Playbookova (bez toga je potrebno koristiti cronjob ili slične alate) te nam daje veoma pregledan način za upravljanje Playbookovima i vizualizaciju podataka.



Slika 19: Pregled AWX Dashboarda [autorski rad]

3.2.8. Ostalo

Ansible ima mnogo mogućnosti za upravljanje konfiguracijom, ali jedna od glavnih mogućnosti koja nedostaje jest automatsko usklađivanje različitih kontrolnih čvorova. Moguće je postaviti više kontrolnih čvorova, ali za njihovo automatsko usklađivanje je potrebno koristiti Ansible Automation Platform (AAP). AAP je iznimno moćan alata za upravljanje cijelom IT infrastrukturom, a Ansible i Ansible Automation Controller (komercijalna verzija AWX-a) su samo

neki od njegovih dijelova. Više detalja o Ansible Automation Platform možete pronaći na Red Hat-ovim stranicama:
`https://www.redhat.com/en/technologies/management/ansible` i
`https://developers.redhat.com/products/ansible/overview`.

3.3. SaltStack

SaltStack (često se naziva i Salt) je na prvu relativno sličan ostalim popularnim alatima za upravljanje konfiguracijom poput Ansible, ali SaltStack nudi neke puno naprednije mogućnosti (ili barem bolju podršku za njih). SaltStack se uglavnom koristi na način da server (master) upravlja klijentima (minionima) na kojima je instaliran klijentski softver, ali SaltStack je jedan od rijetkih alata koji pruža iznimno dobru podršku za bilo kakav način korištenja: klijent i server, samo klijent, samo server i mješavina svega od navedenog u jednom okruženju [41]. U nastavku se očekuje da je na klijentima instaliran klijentski softver (`salt-minion`).

3.3.1. Pojmovnik

SaltStack koristi neke posebne pojmove od kojih je dio vezan uz sol. U nastavku su pojašnjeni neki najbitniji izrazi koji će biti korišteni.

- Formula - skup unaprijed definiranih funkcija za jednostavnije upravljanje popularnim mogućnostima
- Zrna (engl. Grains) - informacije o hardveru i softveru klijenta (memorija, IP adresa, itd.)
- Master - server na koji se spajaju minioni
- Minion - klijentska računala
- Moduli (engl. Modules) - funkcije koje SaltStack izvršava
- Pillar - Skup informacija o grupi klijenata (mogu sadržavati i osjetljive podatke kao npr. lozinke)
- Stanja (engl. States) - opisuju stanja u kojima se treba nalaziti određeni klijent (npr. određuju da mora biti instaliran i pokrenut apache server)

3.3.2. Spajanje SaltStack Miniona na Master

Povezivanje miniona i mastera je iznimno jednostavno. Prvo je potrebno na minionu u datoteci `/etc/salt/minion` maknuti komentar za liniju (znak #) i promijeniti `master: salt` tako da umjesto "salt" piše naziv ili IP adresa mastera. Druga mogućnost je kreirati datoteku `/etc/salt/minion.d/master.conf` u kojoj je prva linija `master:` i IP adresa ili naziv mastera. Također je moguće navesti popis više mastera ili funkciju čija će povratna vrijednost biti IP adresa ili naziv mastera. Moguće je, ali nije nužno definirati ID miniona na sličan

način: micanjem komentara i dodavanjem vrijednosti kod `id:` u `/etc/salt/minion` ili to isto unijeti u novu datoteku `/etc/salt/minion.d/id.conf`.

Nakon toga je potrebno na klijentu pokrenuti naredbu `salt-minion -d` (potencijalno su potrebna prava root korisnika) koja će pokrenuti proces u pozadini i pokušati uspostaviti vezu s masterom. To radi tako što masteru pošalje nasumično generirani ključ koji master može potvrditi ili odbaciti. Na masteru je moguće pokrenuti naredbu `salt-key` koja prikazuje trenutno stanje svih ključeva na masteru. Ako je sve uspješno odrađeno, pod "Unaccepted Keys:" bi trebala biti navedena jedna stavka (u slučaju da smo prije specificirali ID miniona, ta stavka bi trebala biti taj ID). Nakon toga treba pokrenuti naredbu `salt-key -a <key>` gdje je `<key>` odabrana stavka iz prethodne naredbe ili jednostavno `salt-key -A` za prihvaćanje svih još neprihvaćenih ključeva. Master i minion su sada uspješno povezani.

3.3.3. Osnovni primjeri

Svi alati za upravljanje konfiguracijom su na površini relativno slični. Tako se i Salt može koristiti ručnim ili automatskim pokretanjem naredbi s podacima iz datoteka ili ručnim naredbama. Najjednostavnija naredba koja ujedno služi za provjeru dostupnih miniona je `salt '*' test.ping` koja vrača naziv (ID) miniona i True/False ovisno o tome je li uspješno izvršena naredba.

Nas najviše zanimaju stanja. Najosnovniji način za upravljanje stanjima jest koristeći funkciju `apply` iz `state` modula i Salt State (SLS) formule. Slijedi primjer jednostavne SLS formule naziva `ssh.sls` koja provjerava samo je li na minionu instaliran OpenSSH paket. Potrebno je na masteru kreirati datoteku `/srv/salt/ssh.sls` i ona bi trebala izgledati ovako:

```
openssh:
  pkg.installed
```

Ako je OpenSSH paket već instaliran, a najvjerojatnije jest, rezultat bi trebao izgledati ovako ("Started" i "Duration" naravno neće biti identični):

```
      ID: openssh
Function: pkg.installed
  Result: True
 Comment: All specified packages are already installed
 Started: 20:55:48.423204
Duration: 1519.139 ms
 Changes:
```

Slika 20: Stanje openssh paketa [autorski rad]

Slijedi jednostavan Access Control List (ACL) primjer gdje se korisniku "username" eksplicitno daju dozvole čitanja i pisanja za datoteku `/etc/sudoers`.

```
sudoers_acl:
  acl.present:
```

```
    - name: /etc/sudoers
    - acl_type: user
    - acl_name: username
    - perms: rw-
```

3.3.4. Malo naprednije mogućnosti

Ispod se nalazi malo napredniji primjer gdje se provjerava stanje i konfiguracija apache2 servisa na računalu. Prije pokretanja, na masteru je potrebno kreirati ili kopirati ispravnu datoteku `httpd.conf` (i po potrebi direktorij) na lokaciji `/srv/salt/apache2/httpd.conf`.

```
apache2:
  pkg.installed: []
  service.running:
    - enable: True
    - require:
      - pkg: apache2
    - watch:
      - file: httpd_config

httpd_config:
  file.managed:
    - name: /etc/apache2/httpd.conf
    - source: salt://apache2/httpd.conf
    - mode: 644
    - user: root
    - group: root
```

Prvo se provjerava je li paket apache2 instaliran. Zagrade "[]" su u ovom slučaju potrebne i inače se ih preporuča koristiti kako bi se eksplicitno navelo da `pkg.installed` nema nikakve dodatne parametre. Nakon toga se provjerava je li apache2 servis pokrenut i hoće li se pokretati prilikom svakog pokretanja računala (`enable`). Opcija `require` znači da će se navedena provjera servisa odraditi tek u slučaju da je paket apache2 prisutan. Opcija `watch` se poziva u slučaju da je došlo do izmjena. U našem slučaju promjene su ako je paket apache2 instaliran i/ili servis pokrenut. U tom slučaju se poziva `httpd_config` koji kopira datoteku `httpd.conf` s mastera na lokaciji navedenoj pod `source` (`salt://` gleda datoteke od putanje `/srv/salt/`) na minion u lokaciju navedenu kod `name` s navedenim pravima pristupa. U ovom slučaju se kopira konfiguracijska datoteka za čiju je primjenu potrebno resetirati servis, što `watch` radi automatski.

SaltStack uz standardne mogućnosti (korištenje predložaka, grupacija klijenata, uvjetna grananja, izvršavanje naredbi ljuske, korištenje varijabli, itd.) nudi i iznimno napredne mogućnosti kao što su: konstantna kontrola procesa, međusobna komunikacija miniona, obavijesti u slučaju promjene određenih datoteka i još mnoge druge, ali prethodno navedeni primjeri su dovoljni za rješavanje velike većine zadataka.

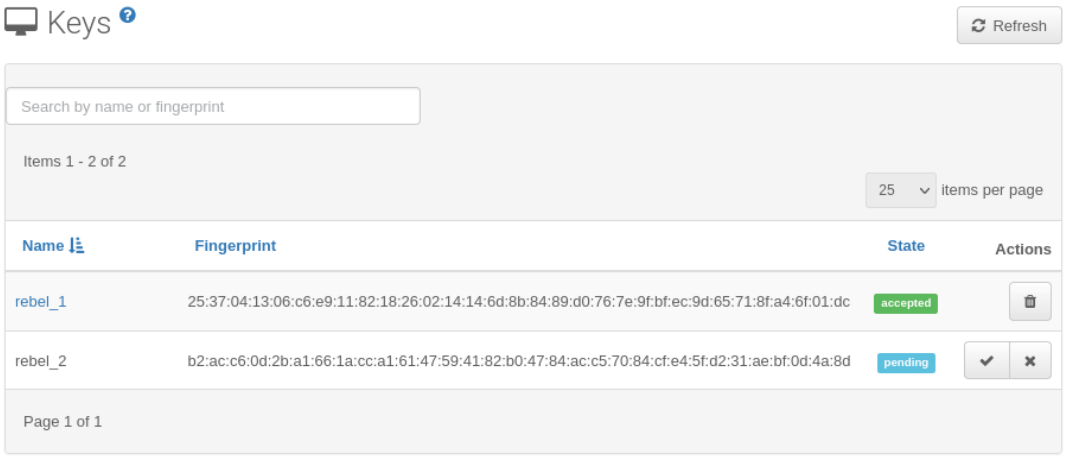
3.3.5. Uyuni (SUSE Manager)

Spacewalk je (do 2020.) bio projekt otvorenog koda na kojem se baziralo komercijalno rješenje za upravljanje infrastrukturom Red Hat Satellite [42]. SUSE Manager je također jedan komercijalni projekt koji se bazirao na Spacewalk projektu, ali je SUSE 2018. uradio fork nad Spacewalk projektom i stvorio Uyuni projekt na kojem se baziraju daljnje verzije alata SUSE Manager [43].

Iako je baziran na alatu za upravljanje infrastrukturom, Uyuni nudi razne mogućnosti i pregledan GUI za upravljanje konfiguracijom koristeći SaltStack. Velik fokus na integraciju sa SaltStack je dobro vidljiv po naziv "Uyuni" koji se odnosi na "Salar de Uyuni", najvećoj slanoj ravnici na svijetu [43].

3.3.5.1. Upravljanje ključevima

U poglavlju 3.3.2 objašnjeno je kako se minioni povezuju na master pomoću ključeva. Uyuni nudi jednostavan frontend za bolji pregled i lakše upravljanje tim ključevima.

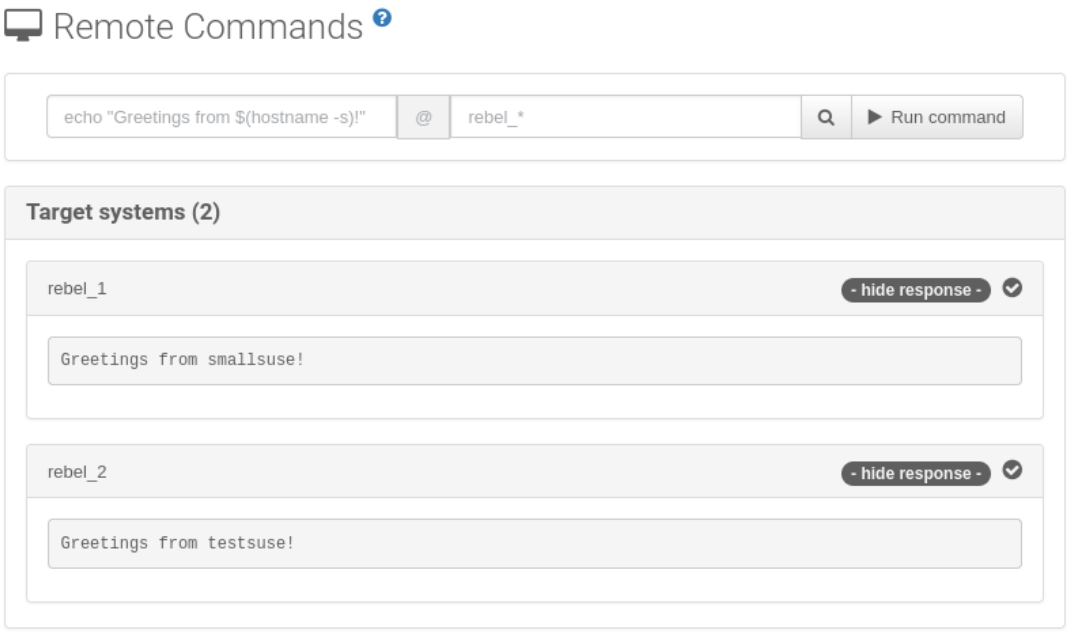


Slika 21: Pregled ključeva sa Uyuni [autorski rad]

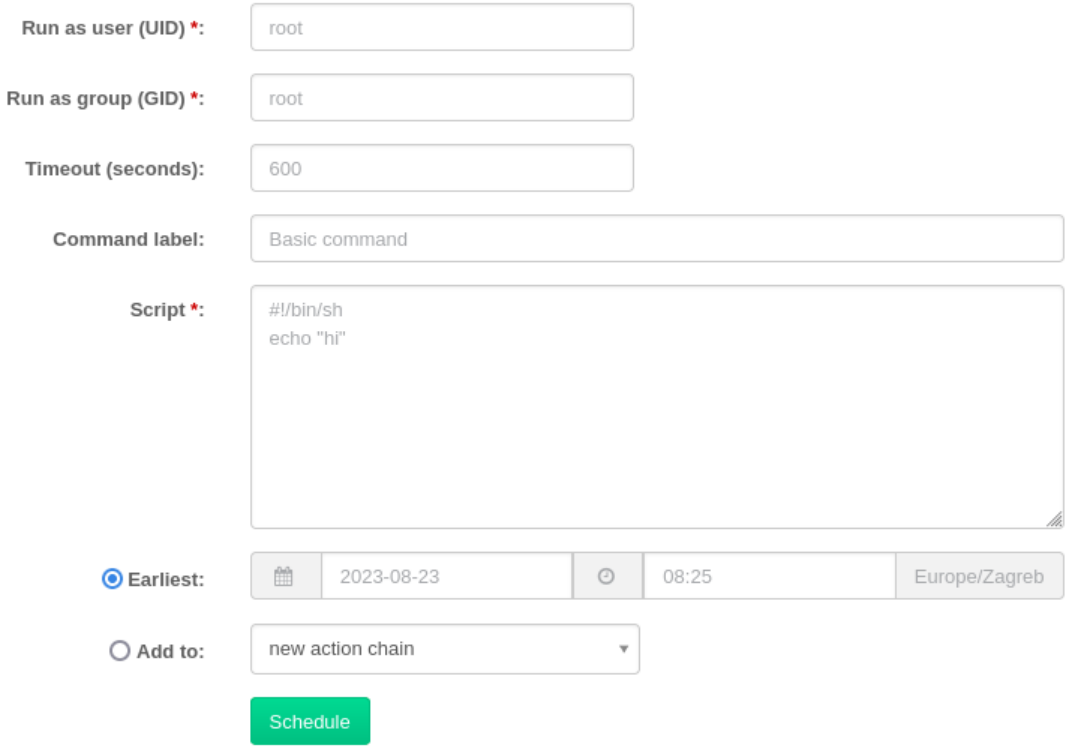
3.3.5.2. Slanje naredbi i raspored

Uyuni pruža jednostavniji način slanja naredbi na minione i pregled njihovih odgovora. Ispod je primjer jednostavne naredbe. Na slici 22 je dobro vidljivo kako je koji dio grafičkog sučelja povezan s prikladnim dijelom naredbe za slanje naredbi putem konzole.

Uyuni također nudi postavljanje rasporeda za izvršavanje određenih naredbi i upravljanje konfiguracijom. To je najbolje vidljivo kod prozora za slanje naredbi na jednog miniona na slici 23.



Slika 22: Slanje naredbi na minione sa Uyuni [autorski rad]



Slika 23: Postavljanje rasporeda naredbe za jedan minion sa Uyuni [autorski rad]

Pregled rasporeda moguće je vidjeti pod "Schedule" gdje je pregled svih uspješno i neuspješno izvršenih kao i još ne izvršenih akcija i njihovi detalji. Detalji o izvršavanju naredbe i vraćenim vrijednostima za pojedinog miniona mogu se pronaći kod "Events" kartice miniona. Postoji još mnogo takvih i sličnih mogućnosti za pregled svih potrebnih informacija o stanju svih miniona, ali ove su najbitnije za upravljanje konfiguracijom.

4. Directory-as-a-Service (DaaS)

Kako servisi u oblaku postaju sve popularniji, nije čudo da postoji i servis u oblaku koji omogućuje upravljanje sigurnosnim politikama. Iako naziv "Directory-as-a-Service" na prvi pogled možda daje dojam da se radi samo o nekom direktoriju na oblaku za spremanje korisničkih podataka, dio servisa pruža ne samo mogućnost upravljanja identitetom, već i nekim politikama. Treba pripaziti da se skraćenica "DaaS" za "Directory-as-a-service" ne miješa s identičnom skraćenicom koja stoji za "Desktop-as-a-service". U ovom radu se "DaaS" koristi isključivo za "Directory-as-a-service".

4.1. JumpCloud

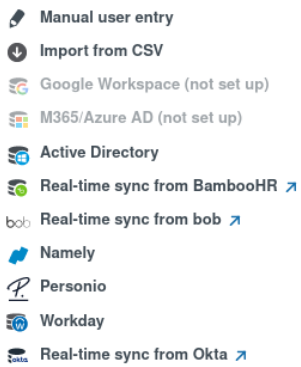
Jedan od najpopularnijih Directory-as-a-Service proizvoda je JumpCloud koji nudi razne mogućnosti upravljanja Windows, Mac i Linux sustavima. Naravno da je u ovom radu fokus na Linux sustave. Podržane Linux distribucije su [44]: Amazon Linux, CentOS, Debian, Fedora, Linux Mint (Cinnamon), Pop!_OS, RHEL, Rocky Linux i Ubuntu.

4.1.1. Dodavanje računala

Dodavanje novog računala je iznimno jednostavno. Na kartici "Devices" treba kliknuti veliki "+" i izabrati odgovarajući operacijski sustav. Za Linux se prikaže jednostavna `curl` naredba koju je potrebno kopirati i pokrenuti na računalu. Ta naredba instalira JumpCloud agent i pokreće "jcagent" servis. To je sve što je potrebno. Računalo je sada uspješno povezano.

4.1.2. Korisnici

JumpCloud nudi razne mogućnosti za uvoz i sinkronizaciju korisničkih računa.



Slika 24: Opcije za dodavanje korisnika na JumpCloud [autorski rad]

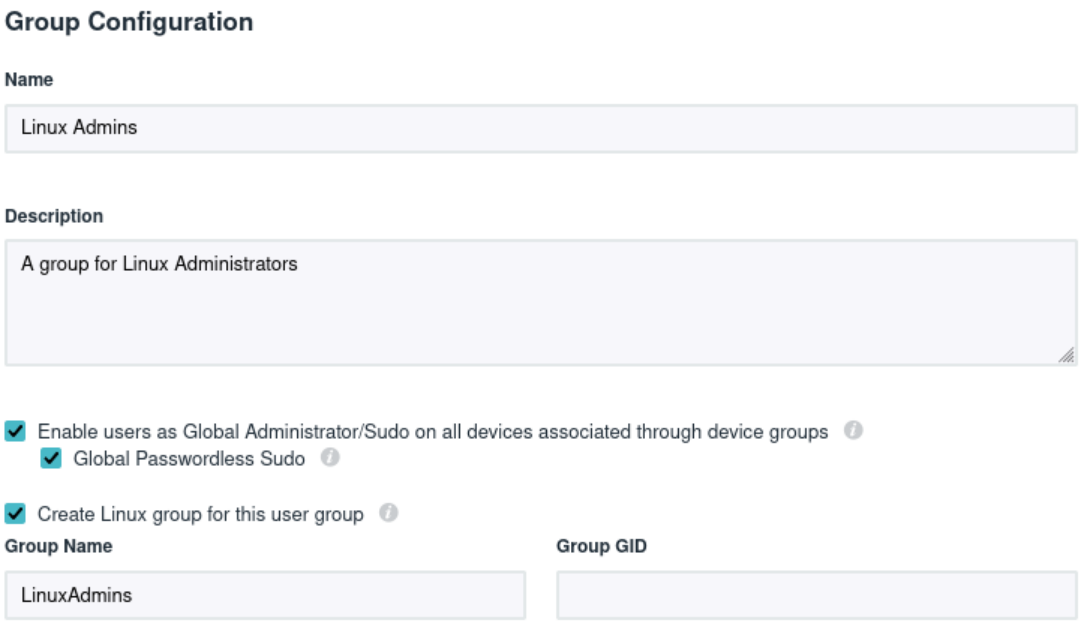
Za korisnike je moguće definirati sve standardne postavke (ime, prezime, korisničko ime, E-mail, lozinka, itd.), ali i napredne (povezivanje korisnika s postojećim lokalnim korisničkim računom na nekom računalu, davanje administratorskih prava za sva računala, obvezna promjena lozinke nakon prve prijave, itd.).

4.1.3. Lozinke

JumpCloud omogućuje globalnu konfiguraciju politika lozinki. Opcije su: minimalna duljina, obavezno sadržavati velika/mala slova, broj ili posebni znak i smije li šifra sadržavati korisničko ime. Moguće je još podesiti povijest lozinki (lozinke iz povijesti se ne mogu ponovo koristiti) i trajanje lozinke u danima (kada je potrebno kreirati novu lozinku). Navedene opcije su globalne i nije ih moguće podesiti za pojedine korisnike, grupe ili računala.

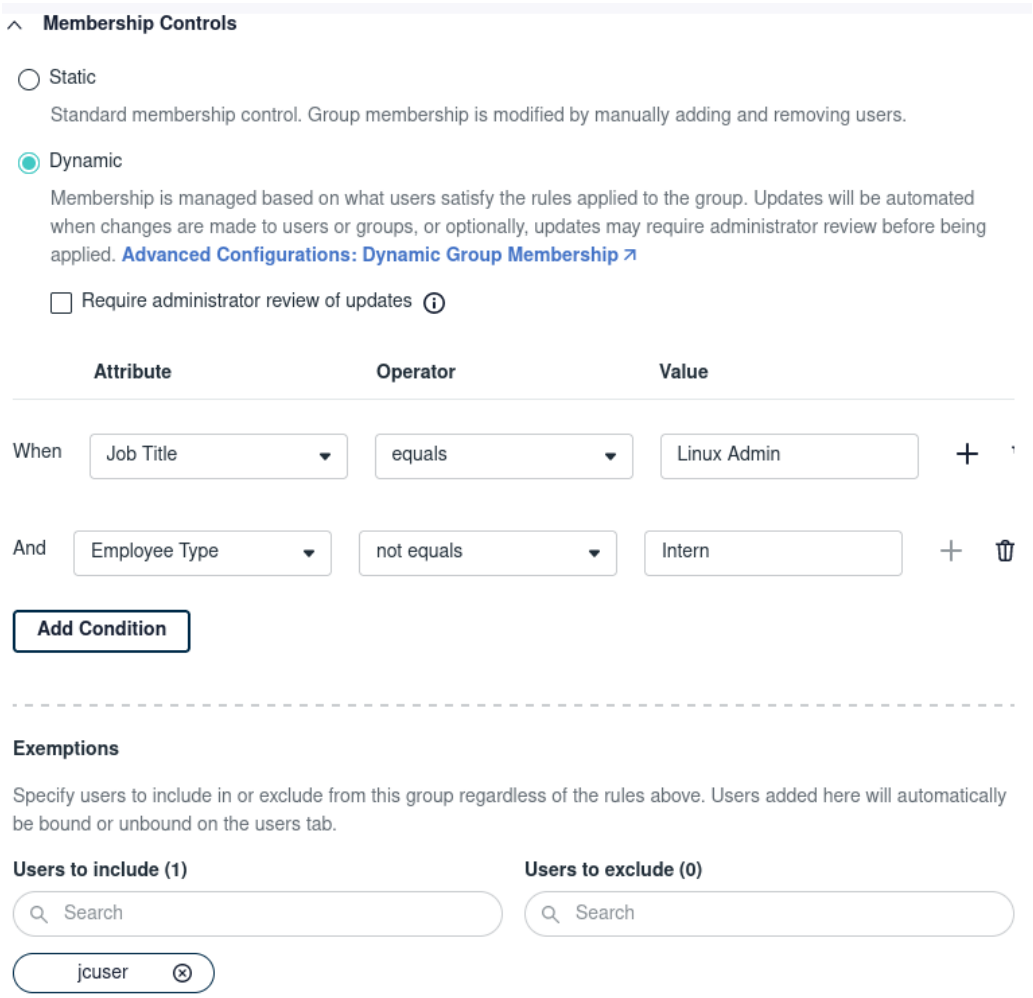
4.1.4. Grupe

Moguće je kreirati grupe korisnika i računala te za korisnike jednostavno podesiti osnovne postavke o `sudo` pravima i lokalnim Linux grupama.



Slika 25: Stvaranje nove grupe korisnika na JumpCloud [autorski rad]

Korisnici mogu biti eksplicitno dodani u grupu, ali i dinamički pomoću pravila. Za grupe računala je gotovo identično, samo su drukčije opcije za pravila (operacijski sustav, arhitektura, verzija, itd.). Kod dinamičkog dodavanja postoji opcija da administratori svejedno moraju prvo potvrditi promjene članova.



Slika 26: Pravila i iznimke za dinamičko dodavanje korisnika u grupu [autorski rad]

4.1.5. Upravljanje politikama

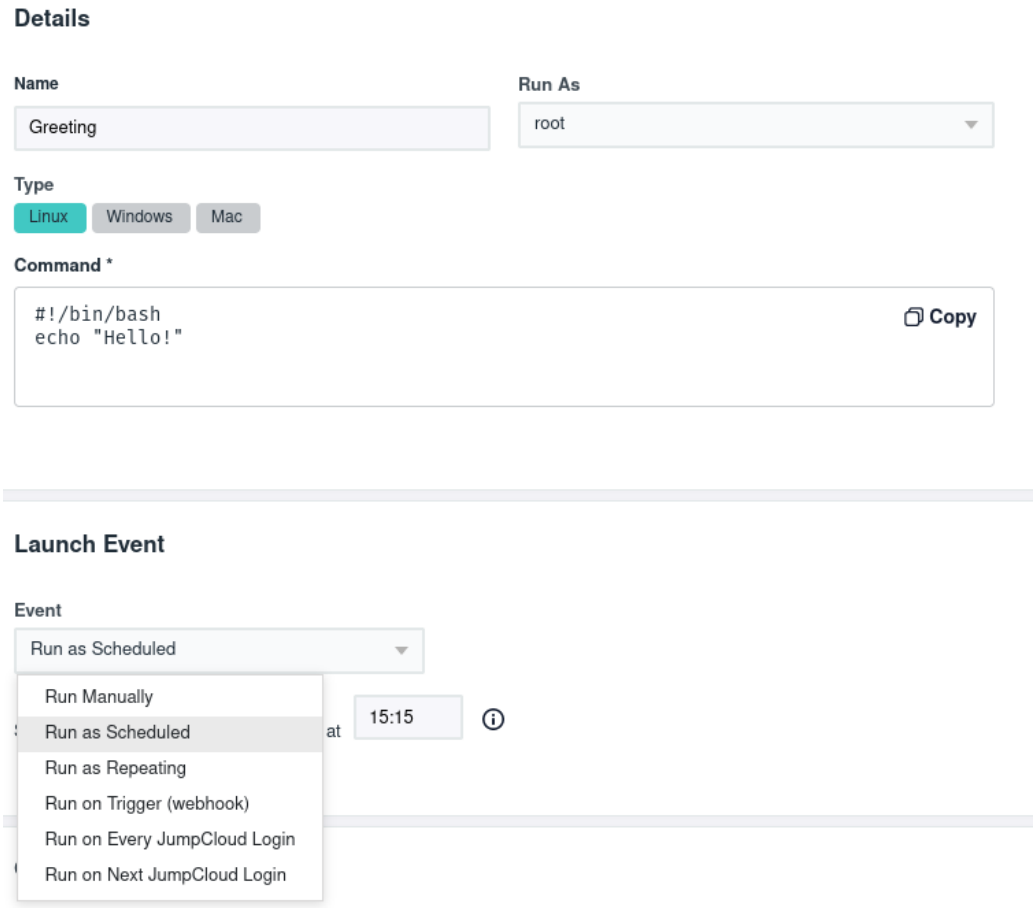
JumpCloud ima ugrađene neke politike za pojedine operacijske sustave, a za Linux trenutno postoji najskromniji izbor, a neke od glavnih su:

- Provjera stanja "Full-Disk" i "Home-Directory" enkripcije
- Konfiguracija `iptables` pravila
- Zabrana korištenja USB uređaja za pohranu podataka
- Sinkronizacija vremena
- Upravljanje dozvolama pristupa nekih datoteka iz `/etc` i `/boot` direktorija
- Zabrana određenih servisa
- Podešavanje vremena neaktivnosti do zaključavanja zaslona
- Postavke za Secure Boot
- Postavke za SSH

4.1.6. Ostalo

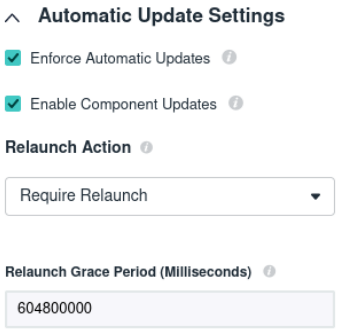
Uz navedene mogućnosti JumpCloud također nudi jednostavnu mogućnost za pravovremeno udaljeno zaključavanje, ponovo pokretanje i gašenje računala te brisanje svih podataka s računala.

Postoji i nekoliko opcija za izvršavanje naredbi na klijentima: prema rasporedu, ponavljajuće, nakon okidača i uz prijavu na JumpCloud.



Slika 27: Opcije za pokretanje naredbi na klijentima [autorski rad]

Moguće je i upravljanje postavkama ažuriranja podržanih operacijskih sustava i web preglednika na klijentima (trenutno samo Chrome).



Slika 28: Opcije za upravljanje postavkama ažuriranja preglednika Chrome [autorski rad]

5. Zaključak

Ovaj rad se bavio detaljnim pregledom mogućnosti koje pružaju neki od najpopularnijih alata za upravljanje sigurnosnim politikama na operacijskom sustavu Linux. Opisan je problem koji je potrebno riješiti i postepeno razvijeni načini rješavanja tog problema koji vode do današnjih rješenja. Ta rješenja za operacijski sustav Linux su često otvorenog koda i sastoje se od više manjih alata koji su navedeni i objašnjeni. Upravljanje sigurnosnim politikama na operacijskom sustavu Linux se često dijeli na upravljanje identitetom i upravljanje konfiguracijom. Izbor sveobuhvatnih alata za upravljanje identitetom s naglaskom na operacijski sustav Linux je relativno malen, ali postojeći alati su dobro razvijeni i pružaju sve potrebne mogućnosti.

Red Hat IdM i njegova besplatna verzija FreeIPA su se pokazali kao najcjelovitija Linux verzija onoga što Microsoft Active Directory omogućuje za Windows. Univention Corporate Server također nudi razne mogućnosti, ali posvećuje više pažnje na inteoperabilnost Windows i Linux sustava i nudi dodatne aplikacije za proširenje mogućnosti servera izvan domene upravljanja identitetom, što je u nekim okruženjima veoma korisno.

Upravljanje konfiguracijom je povezana, ali potpuno različita domena čija je svrha biti Linux zamjena za Group Policy na Windows računalima. Postojeći alati za upravljanje konfiguracijom su na prvi pogled veoma slični jedni drugima, ali se razlikuju u načinu korištenja i naprednim mogućnostima. Ansible je popularan, iznimno jednostavan i dobro integriran s Red Hat-ovim proizvodima, dok SaltStack pruža veliku fleksibilnost u korištenju i neke iznimno napredne mogućnosti uz dodatnu složenost.

Rješenja u oblaku postoje i prilično su jednostavna i inovativna, ali imaju manjak mogućnosti u odnosu na ostala rješenja. Iako omogućuje upravljanje nekim politikama i slanje naredbi na klijente, JumpCloud nije zamjena za alate za upravljanje konfiguracijom niti je potpuna zamjena za Linux-first rješenja. Na primjer, JumpCloud omogućuje davanje `sudo` prava određenim korisnicima i grupama, ali ne i detaljniju konfiguraciju tih prava kao što to omogućuje Red Hat IdM/FreeIPA.

Zaključak je da trenutno na tržištu postoje dobri alati za efektivno upravljanje sigurnosnim politikama na operacijskom sustavu Linux s raznim mogućnostima i razinama složenosti. Vaše specifične potrebe ovise o tome što ćete izabrati, ali izgleda da za gotovo svaku situaciju postoji prikladno rješenje.

Popis literature

[1] „pam(8) - Linux man page.” https://linux.die.net/man/8/pam. (pristupano 1.8.2023.).

[2] „pam_mkhomedir(8) - Linux man page.” https://linux.die.net/man/8/pam_mkhomedir. (pristupano 1.8.2023.).

[3] „nss(8) - Linux man page.” https://linux.die.net/man/5/nss. (pristupano 1.8.2023.).

[4] „SSSD Architecture.” 19.7.2021. https://sssd.io/docs/architecture.html. (pristupano 1.8.2023.).

[5] „Setting up Samba as an Active Directory Domain Controller.” https://docs.ansible.com/ansible/latest/cli/ansible-pull.html. (pristupano 20.8.2023.).

[6] C. Hertel. „Samba: An Introduction, What Samba Does.” 27.11.2021.. https://docs.ansible.com/ansible/latest/cli/ansible-pull.html. (pristupano 20.8.2023.).

[7] B. Atkisson. „Red Hat Identity Manager: Part 1 - Overview and Getting started.” 29.4.2016. https://developers.redhat.com/blog/2016/04/29/red-hat-identity-manager-part-1-overview-and-getting-started. (pristupano 27.7.2023.).

[8] „Red Hat Identity Management.” https://access.redhat.com/products/identity-management/. (pristupano 2.8.2023.).

[9] „FreeIPA.” https://www.freeipa.org/. (pristupano 14.8.2023.).

[10] „SSSD.” https://github.com/SSSD/sssd. (pristupano 14.8.2023.).

[11] „Central management of subordinate user and group ids.” https://freeipa.readthedocs.io/en/latest/designs/subordinate-ids.html. (pristupano 16.8.2023.).

[12] „Univention, References.” https://www.univention.com/references/. (pristupano 14.8.2023.).

[13] „How Ansible works.” https://www.ansible.com/overview/how-ansible-works. (pristupano 15.8.2023.).

[14] „puppet.” https://github.com/puppetlabs/puppet. (pristupano 15.8.2023.).

[15] F. Webber. „The Ruby Behind Chef.” 22.9.2016. https://www.chef.io/blog/the-ruby-behind-chef. (pristupano 15.8.2023.).

[16] „The Ruby Behind Chef.” https://docs.saltproject.io/salt/user-guide/en/latest/topics/overview.html. (pristupano 15.8.2023.).

[17] „YAML Syntax.” https://docs.ansible.com/ansible/latest/reference_appendices/YAMLSyntax.html. (pristupano 15.8.2023.).

[18] L. Kanies. „Puppet Language: Why Puppet Has Its Own Configuration Language.” 3.8.2020.https://www.puppet.com/blog/puppet-language. (pristupano 15.8.2023.).

[19] „Chef - Plain Ruby with Chef DSL.” https://www.tutorialspoint.com/chef/chef_plain_ruby_with_dsl.htm. (pristupano 15.8.2023.).

[20] „Using Jinja with Salt.” https://docs.saltproject.io/salt/user-guide/en/latest/topics/jinja.html. (pristupano 15.8.2023.).

[21] „Templating (Jinja2).” https://docs.ansible.com/ansible/latest/playbook_guide/playbooks_templating.html. (pristupano 15.8.2023.).

[22] „Templates.” https://www.puppet.com/docs/puppet/7/lang_template. (pristupano 15.8.2023.).

[23] „template Resource.” 6.1.2023. https://docs.chef.io/resources/template/. (pristupano 15.8.2023.).

[24] „Glossary.” https://docs.ansible.com/ansible/latest/reference_appendices/glossary.html. (pristupano 15.8.2023.).

[25] K. Sen. „Imperative Models for Configuration Management: Which Is Really Better?” 20.4.2022. https://www.upguard.com/blog/declarative-vs-imperative-models-for-configuration-management. (pristupano 15.8.2023.).

[26] „Ordering States.” https://docs.saltproject.io/en/latest/ref/states/ordering.html. (pristupano 15.8.2023.).

[27] „Learning Ansible basics.” 21.6.2022. https://www.redhat.com/en/topics/automation/learning-ansible-tutorial. (pristupano 16.8.2023.).

[28] „Installing and configuring agents.” https://www.puppet.com/docs/puppet/8/install_agents.html. (pristupano 16.8.2023.).

[29] „Chef Infra Client Overview.” 12.5.2023. https://docs.chef.io/chef_client_overview/. (pristupano 16.8.2023.).

[30] „Understanding SaltStack.” https://docs.saltproject.io/en/getstarted/system/communication.html. (pristupano 16.8.2023.).

[31] „How Puppet Works.” 2018. https://docs.oracle.com/cd/E53394_01/html/E77676/gqqvw.html. (pristupano 16.8.2023.).

[32] „Chef Infra Overview.” 12.5.2023. https://docs.chef.io/chef_overview/. (pristupano 16.8.2023.).

[33] G. S. Ajith. „Chef Infra Overview.” 3.9.2019. https://gayatrisajith.medium.com/beginner-fundamentals-push-pull-configuration-management-tools-85eff1b41447. (pristupano 16.8.2023.).

[34] „Installing Ansible.” https://docs.ansible.com/ansible/latest/installation_guide/intro_installation.html. (pristupano 16.8.2023.).

[35] „Supported operating systems.” https://www.puppet.com/docs/pe/2023.2/supported_operating_systems.html. (pristupano 16.8.2023.).

[36] „Platforms.” 5.6.2023. https://docs.chef.io/platforms/. (pristupano 16.8.2023.).

[37] „Salt supported operating systems.” https://docs.saltproject.io/salt/install-guide/en/latest/topics/salt-supported-operating-systems.html. (pristupano 16.8.2023.).

[38] „Ansible.” https://github.com/ansible/ansible. (pristupano 18.8.2023.).

[39] „How to build your inventory, Defining variables in INI format.” https://docs.ansible.com/ansible/latest/inventory_guide/intro_inventory.html. (pristupano 18.8.2023.).

[40] „ansible-pull, Description.” https://docs.ansible.com/ansible/latest/cli/ansible-pull.html. (pristupano 18.8.2023.).

[41] „SaltStack Flexibility.” https://docs.saltproject.io/en/getstarted/flexibility.html. (pristupano 22.8.2023.).

[42] „Spacewalk.” https://spacewalkproject.github.io/. (pristupano 23.8.2023.).

[43] D. DeMaio. „Uyuni: Forking Spacewalk with Salt and Containers.” 26.5.2018. https://news.opensuse.org/2018/05/26/uyuni-forking-spacewalk-with-salt-and-containers/. (pristupano 22.8.2023.).

[44] „JumpCloud.” https://jumpcloud.com/support/agent-compatibility-system-requirements-and-impacts. (pristupano 28.8.2023.).

Popis slika

1. Jednostavan Automember primjer [autorski rad] 6
2. Osvježavanje automatskih članstva korisnika [autorski rad] 6
3. Na koje korisnike se odnosi ovo sudo pravilo [autorski rad] 7
4. Na koja računala se odnosi ovo sudo pravilo [autorski rad] 7
5. Na koja naredbe se odnosi ovo sudo pravilo [autorski rad] 8
6. U ime kojih korisnika se smiju koristiti navedene naredbe [autorski rad] 8
7. Dodatne opcije za pokretanje sudo naredbi [autorski rad] 9
8. Definiranje pozicije u redu izvršavanja [autorski rad] 9
9. Jednostavan HBAC primjer [autorski rad] 10
10. Jednostavan HBAC Test [autorski rad] 11
11. Jednostavan SELinux primjer [autorski rad] 12
12. Univention Management Console servera ucs5-1 na mreži test.nat [autorski rad] 14
13. Promjena korisničkih podataka [autorski rad] 15
14. App Center [autorski rad] 16
15. Univention Configuration Registry [autorski rad] 17

16. Rezultat prvog pokretanja jednostavnog Playbooka [autorski rad] 22
17. Opis rezultata nakon prvog izvršavanja Playbooka [autorski rad] 24
18. Opis rezultata nakon drugog izvršavanja Playbooka [autorski rad] 24
19. Pregled AWX Dashboarda [autorski rad] 26
20. Stanje openssh paketa [autorski rad] 28
21. Pregled ključeva sa Uyuni [autorski rad] 30
22. Slanje naredbi na minione sa Uyuni [autorski rad] 31
23. Postavljanje rasporeda naredbe za jedan minion sa Uyuni [autorski rad] 31

24. Opcije za dodavanje korisnika na JumpCloud [autorski rad] 32
25. Stvaranje nove grupe korisnika na JumpCloud [autorski rad] 33
26. Pravila i iznimke za dinamičko dodavanje korisnika u grupu [autorski rad] 34
27. Opcije za pokretanje naredbi na klijentima [autorski rad] 35
28. Opcije za upravljanje postavkama ažuriranja preglednika Chrome [autorski rad] . 35

Popis tablica

1. Usporedba alata za upravljanje konfiguracijom [autorski rad] 19