UNIVERZA V LJUBLJANI
FAKULTETA ZA RAČUNALNIŠTVO IN INFORMATIKO

Nino Pelko

Evaluacija platforme za virtualizacijo

DIPLOMSKO DELO
VISOKOŠOLSKI STROKOVNI ŠTUDIJSKI PROGRAM PRVE STOPNJE
RAČUNALNIŠTVO IN INFORMATIKA
Mentor: doc. dr. Rok Rupnik

Ljubljana, 2013

Univerza *v Ljubljani*
Fakulteta *za računalništvo in informatiko*

Tržaška 25
1001 Ljubljana, Slovenija
telefon: 01 476 84 11
01 476 83 87
faks: 01 426 46 47
01 476 87 11
www.fri.uni-lj.si
e-mail: dekanat@fri.uni-lj.si

Št. naloge: 00187/2011
Datum: 05.12.2011

Univerza v Ljubljani, Fakulteta za računalništvo in informatiko izdaja naslednjo nalogo:

Kandidat: **NINO PELKO**

Naslov: **EVALUACIJA PLATFORME ZA VIRTUALIZACIJO**
THE EVALUATION OF VIRTUALIZATION PLATFORM

Vrsta naloge: Diplomsko delo visokošolskega strokovnega študija prve stopnje

Tematika naloge:

Proučite ESXi, brezplačno platformo za virtualizacijo strežnikov podjetje VMWARE. Na njej vzpostavite tri virtualne strežnike in na podlagi tega izdelajte primerjavo med brezplačno ESXi platformo in plačljivo VMWARE platformo.

Mentor:

doc. dr. Rok Rupnik

Fakulteta za računalništvo in informatiko • UNIVERZA V LJUBLJANI • 2

Dekan:

prof. dr. Nikolaj Zimic

Izjava o avtorstvu diplomskega dela

Spodaj podpisani Nino Pelko, z vpisno številko 63050073, sem avtor diplomskega dela z naslovom:

Evaluacija platforme za virtualizacijo

S svojim podpisom zagotavljam, da:

- sem diplomsko delo izdelal samostojno pod mentorstvom doc. dr. Roka Rupnika,
- so elektronska oblika diplomskega dela, naslov (slov., angl.), povzetek (slov., angl.) ter ključne besede (slov., angl.) identični s tiskano obliko diplomskega dela in
- soglašam z javno objavo elektronske oblike diplomskega dela v zbirki "Dela FRI".

V Ljubljani, dne 21. marec 2013 Podpis avtorja:

Kazalo

Seznam tabel

Seznam slik

Povzetek

Kot problem diplomskega dela smo si zastavili postavitev sistema, ki bo gostil storitve elektronske pošte, podatkovne baze in gostovanja spletnih strani. Pri postavitvi je bilo poglavitno načelo postaviti tak sistem s čim manj stroški, tako ob postavitvi, kot tudi pri vzdrževanju, zadostiti pa mora tudi vsaj minimalnim varnostnim zahtevam. Ugotovili smo, da je uporaba virtualiziranih strežnikov najbolj optimalna rešitev predvsem zaradi široke izbire brezplačne programske opreme, ki nam je bila na voljo.

Na fizičnem strežniku smo kot osnovo namestili programsko opremo ESXi, kjer gostujejo trije virtualni strežniki. Sistem Vyatta skrbi za varnost in usmerjanje omrežnega prometa, na drugih dveh pa je nameščen operacijski sistemom Ubuntu, kjer delujejo uporabniške storitve. Vključno z ESXi so vsi različice sistema Linux ali uporabljajo njegovo jedro. Pri vzpostavitvi smo se tudi čim bolj držali že dognanih dobrih praks, kot jih priporoča VMware Inc., proizvajalec opreme ESXi.

Po vzpostavitvi smo celoten sistem tudi testirali glede na učinkovitost, varnost in funkcionalnost. Ugotovili smo, da je bila izbira virtualiziranega okolja dobra, saj sistem uspešno izvaja aplikacije in gosti storitve ob nizkih stroških vzdrževanja.

Ključne besede: virtualizacija, stroškovna učinkovitost, brezplačna programska oprema

Abstract

The diploma thesis focuses on building a system, capable of hosting multiple services, primary e-mail, database and web server. Main goal in achieving such system were low development and maintenance costs, while keeping security on a high level. Due to estensive choice of free software available, such system was successfully built using virtualization technology.

VMware's solution ESXi was used on a physical server, which hosted multiple virtualized servers. Security and routing is handled by Vyatta system, while user services are maintained by two servers using Ubuntu operating system. While establishing all those versions of Linux system (or based on its kernel), following best practises were not neglected.

Security, performance in functional tests performed after completion of project were successfully executed, thus proving efficiency of virtualization environment. There has not been any expense for software and the system keeps low running and maintenance costs.

Keywords: virtualization, cost reduction, open-source software

1. Uvod

1.1. Zgodovina virtualizacije

Zgodovina virtualizacije sega v 60. leta 20. stoletja, ko so bili računalniki še v zgodnji fazi razvoja in je razvoj stremel k čim boljši izkoriščenosti. Virtualizacija na teh sistemih je bila prilagojena tako, da je bil osrednji računalnik (ang. mainframe) razdeljen v več virtualnih, ki so vsak zase obdelovali svoj proces. Osrednji računalnik je tako opravljal več procesov hkrati.

Z razvojem so se računalniki pocenili in šele leta 1999 je podjetje VMware, Inc. razvilo sistem, ki je omogočal virtualizacijo računalnikov na arhitekturi x86. Glavna ovira pri tem razvoju je bilo izvajanje 17 strojnih ukazov, ki jih virtualizirana centralna procesna enota ni mogla izvesti brez napak. V VMware so težavo rešili tako, da je vsak tak ukaz sprožil past, ustrezen podprogram pa ga je izvedel na način, ki ga je moč izvajati tudi v virtualnem okolju.

VMware je vse do danes[1] ostal vodilni proizvajalec programskih rešitev za virtualizacijo računalniških okolij, uporabnike njegovih rešitev najdemo kar pri vseh podjetij s seznama Fortune 100 [13]. Drugi ponudniki rešitev sta med drugim Microsoft in Oracle.

Virtualizacija se je izkazala kot odlična rešitev in je v uporabi tako v gospodarstvu kot tudi v javni upravi, z operacijskim sistemom Windows 7 pa jo (brez namestitve dodatne programske opreme) uporabljamo tudi doma. Razvoju sta sledila tudi glavna proizvajalca procesorjev Intel in AMD, ki s svojima tehnologijama VT-x in AMD-V omogočata učinkovitejše izvajanje sistemov za virtualizacijo.

1.2. Opis problema

V našem primeru smo želeli vzpostaviti sistem več strežnikov, ki bodo opravljali naloge gostovanja spletne strani, strežnika za elektronsko pošto in hranjenja podatkov v bazi. Pri izbiri sistema smo upoštevali več dejavnikov, ki so opisani v Tabeli 1.

Dejavnik	Prioriteta
Cena vzdrževanja	1
Cena postavitve	2
Prilagodljivost	3
Stabilnost in varnost	4
Administracija	5
Čas postavitve	6

Tabela 1: Dejavniki pri izbiri sistema

[1] Leta 2012

Iz Tabele 1 je razvidno, da sta ceni vzdrževanja in postavitve imela največjo težo pri izbiri, medtem ko je čas postavitve najmanj pomemben, saj nismo imeli določenega roka. Pri izbiri smo preučili več možnosti, ki so opisane v Tabeli 2. Vsako rešitev smo ocenili glede na dejavnike od 1 do 3, pri čemer je vrednost 3 najboljša, in jo vpisali v levi stolpec vsake rešitve. Vrednost v drugem stolpcu pa predstavlja uteženo vrednost kot zmnožek ustreznosti glede na dejavnik ter prioritete dejavnika od 6 do 1, pri čemer ima najvišja prioriteta vrednost 6.

	Več strežnikov		En strežnik		En strežnik z več virtualnimi	
Cena vzdrževanja	1	6	3	18	2	12
Cena postavitve	1	5	3	15	3	15
Prilagodljivost	2	8	1	4	3	12
Stabilnost in varnost	3	9	1	3	3	9
Administracija	1	2	3	6	1	2
Čas postavitve	1	1	2	2	2	2
Seštevek		**31**		**48**		**52**

Tabela 2: Sistemi na izbiro

Pri izbiri smo vnaprej predvideli, da bomo uporabljali brezplačno programsko opremo, saj primerjava s plačljivo ne bi bila smiselna. Za naš primer se je kot najboljša rešitev pokazal sistem enega strežnika, na katerem vzpostavimo delovanje več virtualnih.

1.3. Prednosti in slabosti virtualizacije

Kot pri vsaki implementaciji določenega sistema tudi pri virtualizaciji obstajajo prednosti in slabosti. Kot je razvidno iz spodnjega seznama je prednosti veliko, kar potrjuje primernost naše odločitve o izbiri sistema iz prejšnje točke.

Prednosti:

- Boljša izkoriščenosti strojne opreme – na računalniku lahko postavimo poljubno število virtualnih strežnikov oz. delovnih postaj, dokler v celoti ne izkoristimo sistemskih sredstev.
- Lažje testiranje sistemov – virtualni strežnik lahko hitro postavimo in tudi hitro odstranimo, kar je zelo uporabno, če še ugotavljamo, kateri operacijski sistem je najbolj primeren za določene potrebe.
- Prenosljivost sistemov – virtualni strežniki so v bistvu le datoteka (oz. več njih), torej jih je mogoče prenašati in kopirati med različnimi gostiteljskimi sistemi. Olajšana je tudi izvedba varnostnih kopij celotnega strežnika, restavriranje pa neprimerno hitrejše kot pri sistemih, ki niso virtualizirani.

- Večja varnost – sistemi za virtualizacijo nam omogočajo tudi postavitev virtualnih omrežij, med katerimi lahko postavimo požarne zidove ter sisteme za zaznavo in preprečevanje vdorov v sistem.
- Nižje tveganje – več strežnikov, pri katerih vsak opravlja svojo nalogo predstavlja manjše tveganje, kot če vse naloge opravlja en sam strežnik. V primeru odpovedi enega izmed njih ostali sistemi lahko še vedno normalno delujejo, če niso odvisni od prizadetega sistema.

Slabosti:

- Obširnejša administracija – virtualizirani sistemi zahtevajo več administracije, saj je poleg gostujočih strežnikov potrebno vzdrževanje tudi gostiteljskega sistema. Oseba, ki s takim sistemom upravlja mora biti ustrezno usposobljena.
- Višje tveganje – čeprav imamo pri virtualizaciji manj tveganja kot pri sistemu z enim fizičnim strežnikom, smo še vedno bolj ranljivi kot pri sistemu z več fizičnimi strežniki. V primeru odpovedi gostiteljskega sistema namreč odpovejo tudi vsi virtualizirani strežniki.
- Usmerjenost k pretirani virtualizaciji – nekatera podjetja stremijo k popolni virtualizaciji, kar pa ni najboljša praksa. V vsakem informacijskem sistemu je priporočljivo obdržati po vsaj en fizični domenski in imenski strežnik.

1.4. Izbira in priprava strojne in programske opreme

Pri izbiri programske opreme za naš sistem nismo imeli veliko izbire, saj smo se omejili zgolj na brezplačne rešitve. V tem segmentu je najprimernejši izdelek ESXi podjetja VMware, ki se v veliki meri uporablja tudi v komercialne namene.

Veliko možnosti smo imeli pri izbiri strojne opreme (strežnika), vendar smo morali upoštevati združljivost. Pri tem nam je na voljo spletna stran podjetja VMware [1], kjer je dostopen seznam strežnikov, primernih za vsak izdelek tega podjetja. Ker smo v našem primeru želeli sistem postaviti na čim cenejši strojni opremi, nakup profesionalnega strežnika ni bila optimalna rešitev. Zato nam je bila v pomoč neuradna stran [7] s seznamom delovnih postaj in druge strojne opreme, ki preverjeno deluje z izdelkom ESXi. Po pregledu smo se odločili za delovno postajo Dell Optiplex 755, saj velja za stabilen sistem, vgrajeni procesor pa podpira tudi tehnologijo VT-x. Podjetje VMware ponuja tudi nekoliko predelano različico izdelka ESXi, ki je prirejena posebej za sisteme proizvajalca Dell. Z uporabo te različice smo se izognili vsem morebitnim težavam z gonilniki za strojno opremo te delovne postaje.

2. Glavni del

2.1. Pregled dobrih praks

Pred postavitvijo programske opreme ESXi, ki smo jo izbrali za gostiteljski sitem, smo preverili priporočila in navodila podjetja VMware. S premišljenim načrtovanjem in izogibanju pogostim napakam se lahko izognemo marsikateri težavi kasneje. V primeru, da imamo zagotovljeno tudi plačljivo podporo za naš sistem, nam proizvajalec pogosto ne želi pomagati, dokler sistema ne popravimo v skladu s temi praksami.

Navodila za postavitev, nadgradnjo, pogosta vprašanja in drugi napotki so objavljeni na spletni strani podjetja VMware [3]. Pred postavitvijo smo si pomagali z dokumentom *Getting Started with ESXi Installable*, ki opisuje začetni postopek postavitve fizičnega strežnika s programsko opremo ESXi ter virtualnih postaj.

Drug pomemben dokument je *Performance Best Practices for VMware vSphere 4.0*, ki smo ga v celoti podrobno preučili pred samim začetkom postavitve, saj opozarja na več morebitnih težav pri postavitvi sistema. Priporočila smo predstavili v naslednjih podpoglavjih.

2.1.1. Strojna oprema

Pri izbiri strojne opreme nam VMware daje več priporočil, opisanih v spodnjih alinejah.

- Temeljit pregled pomnilnika. Pri tem smo uporabili namensko programsko opremo Memtest86, ki testira pomnilnik in preveri ali prihaja do napak pri zapisu in branju podatkov. V primeru zaznane napake gre najverjetneje za okvarjen pomnilniški modul, ki ga je potrebno zamenjati.
- Strojna oprema mora biti združljiva s programsko. Pri izbiri opreme si lahko pomagamo s podatki na spletnih straneh, omenjenimi v poglavju 1.4.
- Procesor (ali več njih) naj podpira napredne tehnologije, ki so v pomoč virtualizaciji. Intelovi je razvil tehnologiji VT-x za lažje izvajanje ukazov ter EPT za naprednejše upravljanje z delovnim spominom. Analogni tehnologiji podjetja AMD se imenujeta AMD-V in RVI (slednja včasih tudi NPT).
- Uporabljen mrežni vmesnik(i) naj bo namenjen za strežnik, uporaba poceni vmesnikov, ki jih srečujemo v delovnih postajah ni priporočljiva. Vsa omrežna oprema (tako aktivna kot pasivna) naj bo ustrezno nastavljena za uporabo najvišjih podprtih hitrosti.
- Izklop vseh nepotrebnih komponent sistema kot so CD/DVD in disketni pogoni, zvočne kartice in podobno.
- VMware daje nekaj priporočil tudi pri strojnih nastavitvah v sistemu BIOS ter nadgradnjo le-tega, če obstaja novejša različica. Najpomembnejša nastavitev je vklop tehnologij v

podporo virtualizaciji, ki smo jih omenili v tretji alineji. Smiselno je izklopiti tudi vse nepotrebne funkcije (npr. vmesniki USB, COM, LPT) in tehnologije za manjšo porabo energije. Te namreč uporabljajo sistemska sredstva, ki tako niso na voljo opremi ESXi ali v celoti zmanjšujejo zmogljivost sistema. Nastavitve se sicer razlikujejo med strežniki, zato je priporočljivo preveriti tudi navodila proizvajalca.

Dokument opisuje tudi priporočila glede diskovnega polja, ki pa je postavljeno ločeno od strežnika in ga v našem sistemu nismo potrebovali.

2.1.2. Operacijski sistemi na virtualnih strežnikih

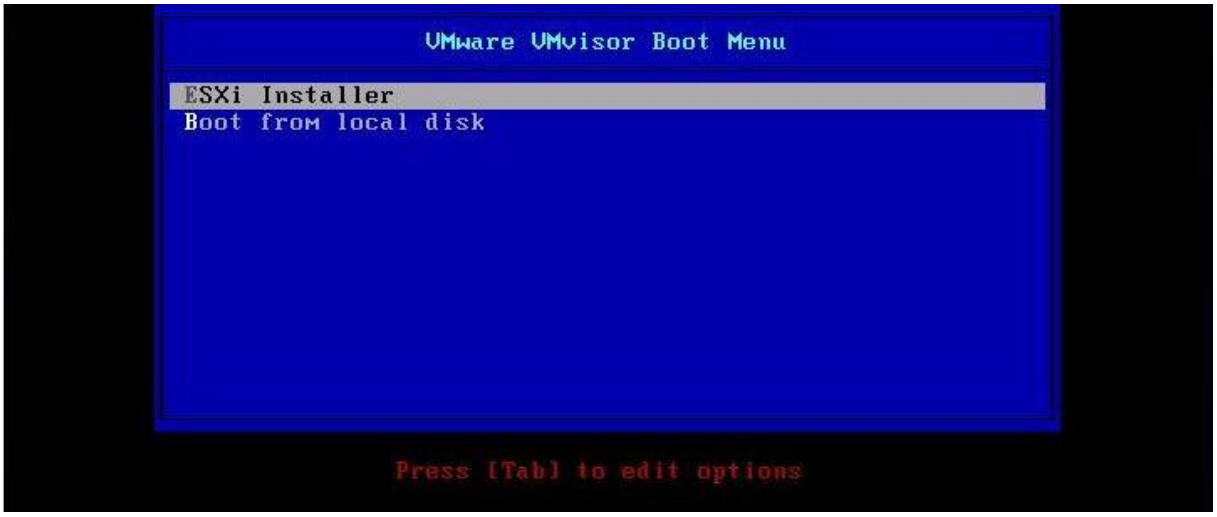
V omenjenem dokumentu več priporočil tudi glede operacijskih sistemov na gostujočih virtualnih strežnikih.

- Priporočljiva je uporaba le podprtih operacijskih sistemov. Seznam teh je na voljo na spletni strani [4] in obsega vse glavne različice sistemov Windows, Linux, Mac OS, OS/2, Solaris in drugih.
- Če je za nameščen operacijski sistem na voljo orodje VMware Tools, ga namestimo.
- Izklopimo ohranjevalnik zaslona in druge olepševalne funkcije, kot so glajenje robov, prosojnost oken in animacije. Tovrstne funkcije namreč po nepotrebnem uporabljajo sistemska sredstva.
- Opravila, ki zahtevajo več sistemskih sredstev ustrezno razporedimo po urniku, da se izvajajo v času manjše obremenitve. Pri tem moramo predvideti tudi porabo sredstev na drugih virtualnih strežnikih, ki gostujejo na istem fizičnem strežniku.
- Za natančno merjenje časa je priporočljiva uporaba protokola NTP ali posebne programske opreme. Tovrstno funkcijo omogoča tudi orodje VMware Tools, vendar naj se le-to uporabi le, če druge rešitve nimamo na voljo.
- Če je možno, za gostujoči operacijski sistem uporabimo tistega, ki podpira tudi tehnologijo VMI (ang. Virtual Machine Interface), ki pripomore k učinkovitosti sistema. Če sistem postavljamo na novejši verziji opreme ESXi to ni nujno smiselno, saj je tehnologija v ukinjanju, njene funkcionalnosti pa proizvajalci procesorjev vgrajujejo v sam čip.
- Dodatne informacije o gostujočih operacijskih sistemih so na voljo v dokumentu *Guest Operating System Installation Guide*.

2.2. Namestitev programske opreme VMware ESXi 4.1

Sama namestitev na strežnik ni bila težavna, kot pri običajnih operacijskih sistemih smo v enoto za medij CD oz. DVD vstavili namestitveni medij ter enoto v sistemu BIOS nastavili kot primarni izvor za zagon operacijskega sistema. Ob naslednjem zagonu nas je pričakal zaslon (Slika 1),

ki nam ponuja namestitev opreme ESXi oz. zagon operacijskega sistema, ki je že na lokalnem trdem disku.



Slika 1: Prvi zaslon pri namestitvi opreme ESXi

Izbrali smo prvo možnost, s čimer se je postopek namestitve pričel. Med potekom nas je namestitveni program povprašal po nekaterih preprostih nastavitvah (izbira trdega diska, strinjanje s pogoji EULA[2], ipd.). Po uspešno končanem postopku nas je pričakal začetni zaslon, od koder smo že lahko upravljali najpomembnejše nastavitve sistema.

2.3. Konfiguracija programske opreme ESXi

Naš sistem lahko upravljamo na več različnih načinov:

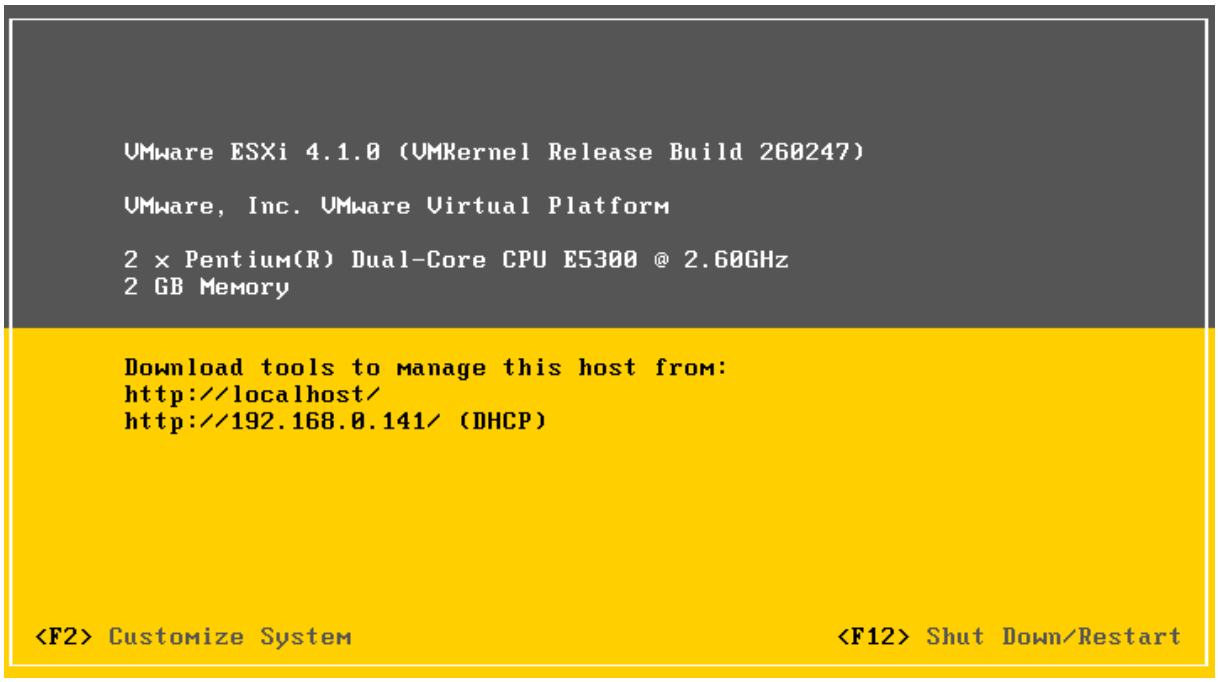
- neposredno preko uporabniškega vmesnika,
- preko oddaljenega dostopa z uporabo protokola SSH in
- z namensko programsko opremo VMware vSphere Client.

Vsak od načinov ima svoje prednosti in slabosti, prikazali pa bomo uporabo vseh treh.

2.3.1. Strežniška konzola

Na primeru (Slika 2) vidimo, da je strežnik že prevzel naslov IP, ki mu ga je dodelil strežnik DHCP oz. mrežni usmerjevalnik, ki ima tak strežnik vgrajen. Taka uporaba ni priporočljiva, saj lahko povzroči nedosegljivost sistemov, če se naslov spremeni. Temu se lahko izognemo z nastavitvijo rezervacij IP naslovov na strežniku DHCP ali z nastavitvijo statičnega naslova IP, kar smo uporabili tudi pri našem sistemu.

[2] Angleška kratica za *End User Licence Agreement*



Slika 2: Začetni zaslon po postavitvi opreme ESXi

Za nastavitev statičnega naslova IP je bilo potrebno vstopiti v nastavitveni meni s tipko F2, ter vpisom uporabniškega imena in gesla. Ob namestitvi geslo ni nastavljeno, uporabniško ime pa je 'root'. Za vstop zato zadošča le potrditev na tipko Enter (Slika 3).

Slika 3: Vnos uporabniškega imena in gesla

V tem trenutku je bil naš sistem zelo ranljiv, saj bi do njega lahko dostopal kdorkoli v našem internem omrežju. Geslo je zato potrebno obvezno nastaviti še preden omogočimo dostop do strežnika večji skupini uporabnikov, kar storimo z izbiro možnosti **Configure Password**.

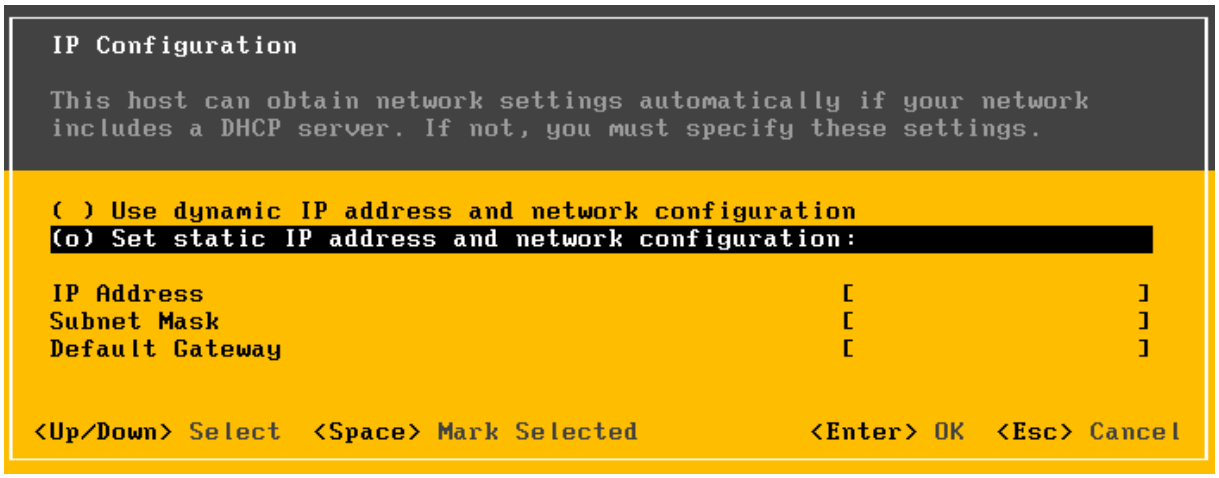
Nastavitev naslova IP smo uredili v drugem meniju in sicer **Configure Management Network**. Tu je na voljo več nastavitev:

- pregled mrežnih vmesnikov,
- nastavitev številke virtualnega omrežja VLAN,
- nastavitev naslova IP,

- nastavitev strežnikov DNS in
- nastavitev domene.

V primeru, da ima strežnik več mrežnih vmesnikov, je možna nastavitev redundance in porazdelitev bremena med dvema ali več vmesniki. V komercialnih sistemih je to priporočljivo, saj s tem zmanjšamo število točk v sistemu, ki ob okvari povzročijo izpad delovanja celotnega sistema (ang. single point of failure).

V našem primeru smo imeli na voljo dva mrežna vmesnika, enega smo uporabili za zunanji (v smeri proti internetu), drugi pa je ostal neizkoriščen in ga lahko v prihodnosti uporabimo za notranji promet (lokalno omrežje). Tako nam je na eni točki v sistemu omogočeno filtriranje in nadzor nad celotnim prometom, v smeri proti ali iz interneta. V našem primeru smo le nastavili požarni zid in sistem za zaznavanje vdorov, kar smo opisali v poglavjih 2.5.3. in 2.5.4. Kljub neizkoriščenosti smo obema mrežnima vmesnikoma ročno dodelili naslov IP (Slika 4). Na zunanjem vmesniku smo nastavili javni naslov IP, ki nam ga je dodelil ponudnik interneta, na notranji strani pa poljuben naslov iz segmenta zasebnih naslovov IP, v našem primeru 10.1.0.3. Pomembni sta tudi nastavitvi maske omrežja in privzetega prehoda. Za zunanji mrežni vmesnik smo morali vpisali podatke, ki nam jih je posredoval ponudnik interneta, pri notranjem morajo nastavitve biti skladne z našo zasnovo omrežja. Za privzeti prehod smo nastavili naslov IP zunanjega mrežnega vmesnika, saj bo čezenj potekal ves promet proti internetu. Za masko omrežja pa smo izbrali kar najpogosteje uporabljano, 24-bitno masko (255.255.255.0), saj ne načrtujemo, da bomo v omrežje dodajali večje število strežnikov in delovnih postaj, če sploh.



Slika 4: Nastavitev naslova IP na mrežnem vmesniku

Naslednja pomembna nastavitev je vpis strežnikov DNS. Navadno ima vsak ponudnik vsaj dva svoja strežnika DNS, ki ju lahko uporabimo. Če tega ne moremo ali želimo storiti, lahko uporabimo nacionalne strežnike DNS (v Sloveniji jih upravlja javni zavod ARNES) ali katere koli

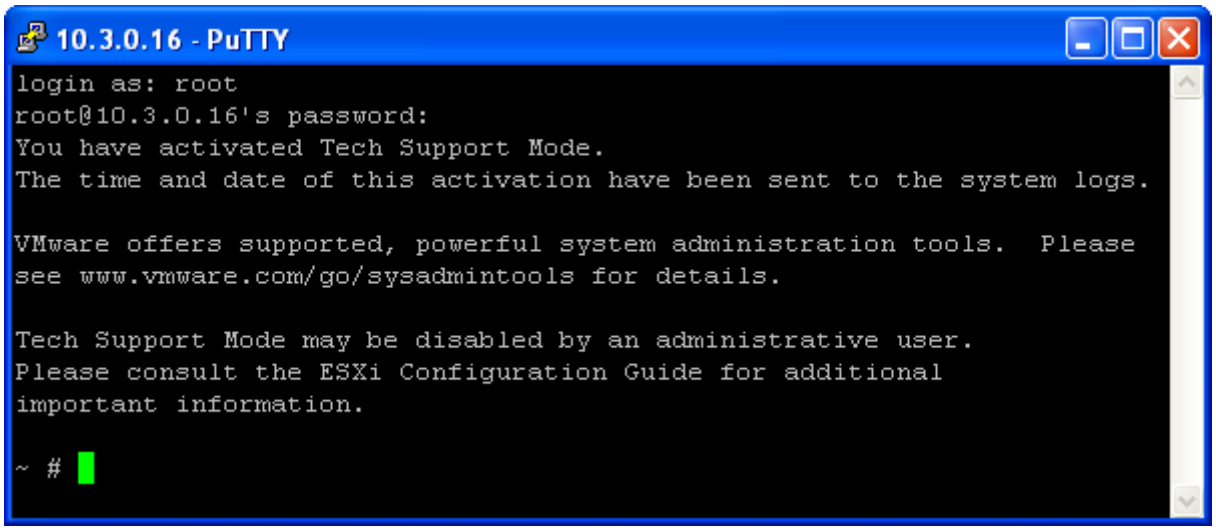
druge javne strežnike. Primer takih sta Googlova strežnika, ki imata[3] naslova IP 8.8.8.8 in 8.8.4.4. Pomembno je, da uporabimo zanesljive strežnike DNS, saj smo sicer izpostavljeni raznovrstnim napadom, še posebej ribarjenju (ang. phishing).

V naš sistem smo vpisali ponudnikova strežnika DNS ter s tem opravili glavne nastavitve na sistemu, ki je sedaj pripravljen na postavitev virtualnih strežnikov. Če bi vseeno želeli poseči globlje v sistem, imamo možnost to storiti s pomočjo protokola SSH, kar je prikazano v naslednjem poglavju.

2.3.2. Protokol SSH

Programska oprema ESXi uporablja jedro operacijskega sistema Linux in omogoča delo tudi preko konzole. Proizvajalec podpore za uporabo protokola SSH sicer ne ponuja, zato je tudi privzeto izklopljen.

Za vstop v konzolo pritisnemo kombinacijo tipk Alt + F2. Sedaj moramo vpisati ukaz *unsupported*, nato pa še geslo, ki smo ga nastavili za uporabnika 'root'. Protokol omogočimo v datoteki `/etc/inetd.conf`, ki jo odpremo z urejevalnikom vi, ter odstranimo znak `#` pred besedilom `ssh` v 32. vrstici datoteke. Protokol po ponovnem zagonu strežnika že deluje in nanj se lahko povežemo z ustreznim odjemalcem (Slika 5).



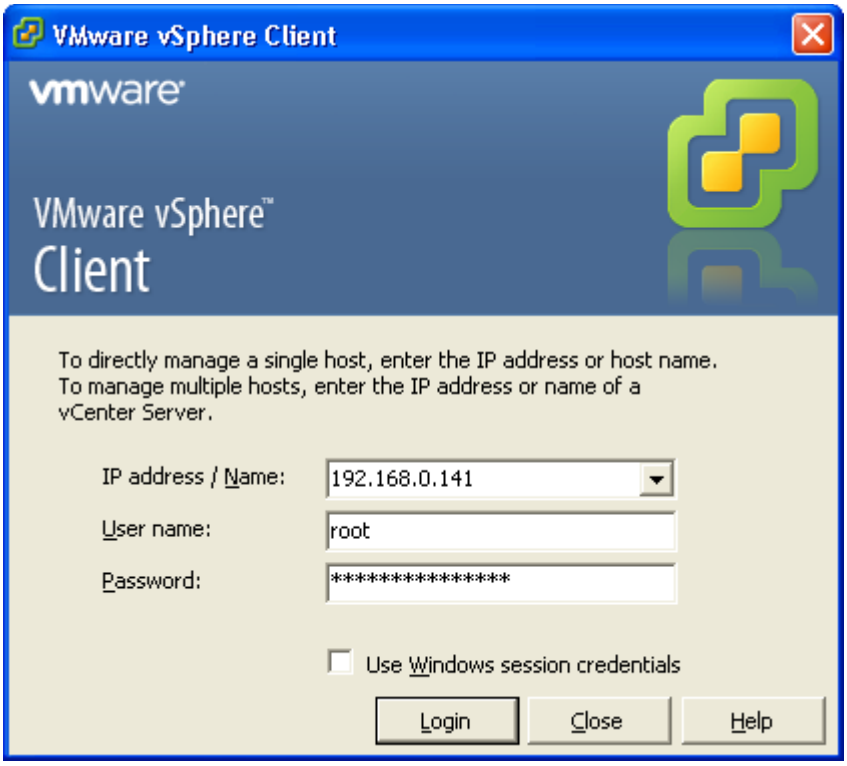
Slika 5: Oddaljen dostop do konzole strežnika ESXi

2.3.3. Namenska aplikacija

Glavno orodje za delo s strežnikom ESXi je VMware vSphere Client. V večjem okolju je priporočljiva uporaba namenskega strežnika VMware vCenter, s katerim lahko upravljamo več strežnikov ESX in ESXi, vSphere Client namreč omogoča le delo z enim.

Za dostop do našega strežnika (Slika 6) potrebujemo naslov IP, uporabniško ime in geslo, ki smo jih določili, kot je opisano v poglavju 2.3.1.

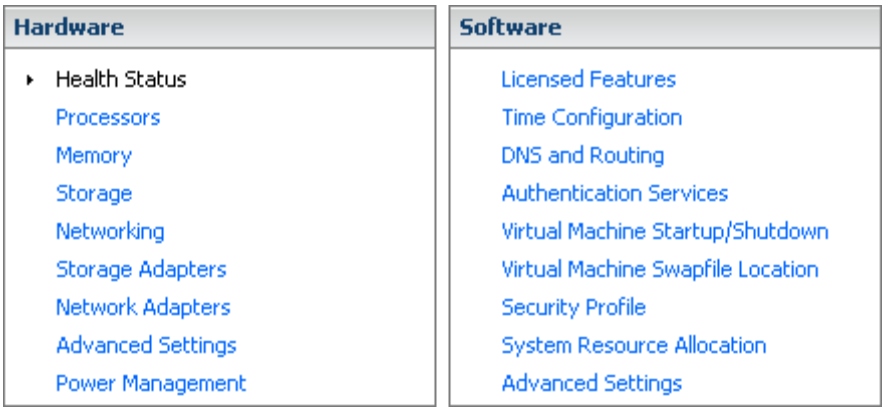
[3] Na dan 31.12.2011



Slika 6: Prijava v aplikacijo VMware vSphere Client

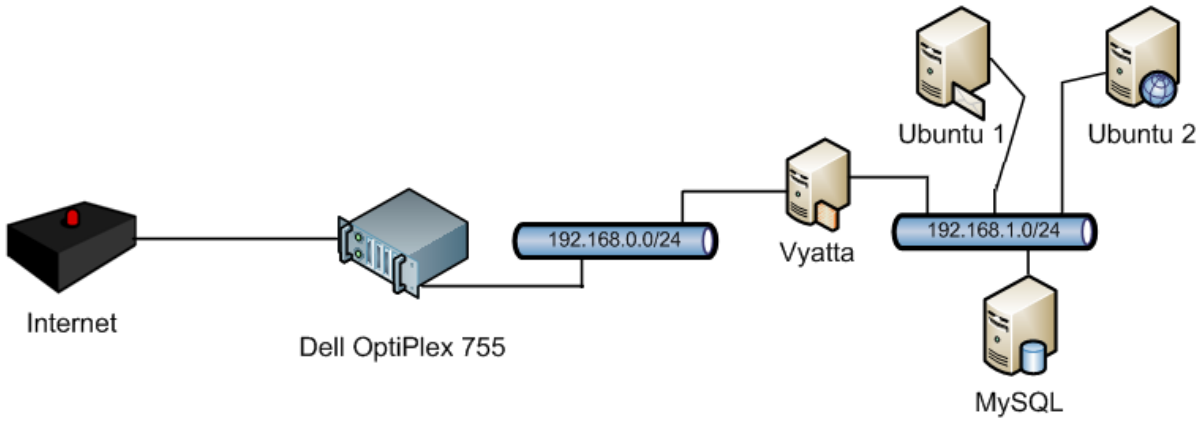
Po uspešni prijavi na strežnik smo lahko pričeli s postavitvijo sistema, kot smo ga predhodno načrtovali. Ustrezno načrtovanje je izjemnega pomena, saj s tem zagotovimo višjo varnost in se izognemo morebitnim kasnejšim spremembam, ki bi zahtevale nedosegljivost strežnikov.

Ker oprema ESXi omogoča izjemno veliko funkcionalnosti je tudi orodje vSphere Client precej obsežno z nastavitvami in funkcionalnostmi za pregled sistema. Glavne nastavitve in upravljanje s sistemom so nam na voljo na zavihku **Configuration** (Slika 7), drugje pa upravljamo s pravicami, spremljamo uporabo sistemskih sredstev ter pregledujemo dnevniške zapise.



Slika 7: Nastavitve na zavihku "Configuration"

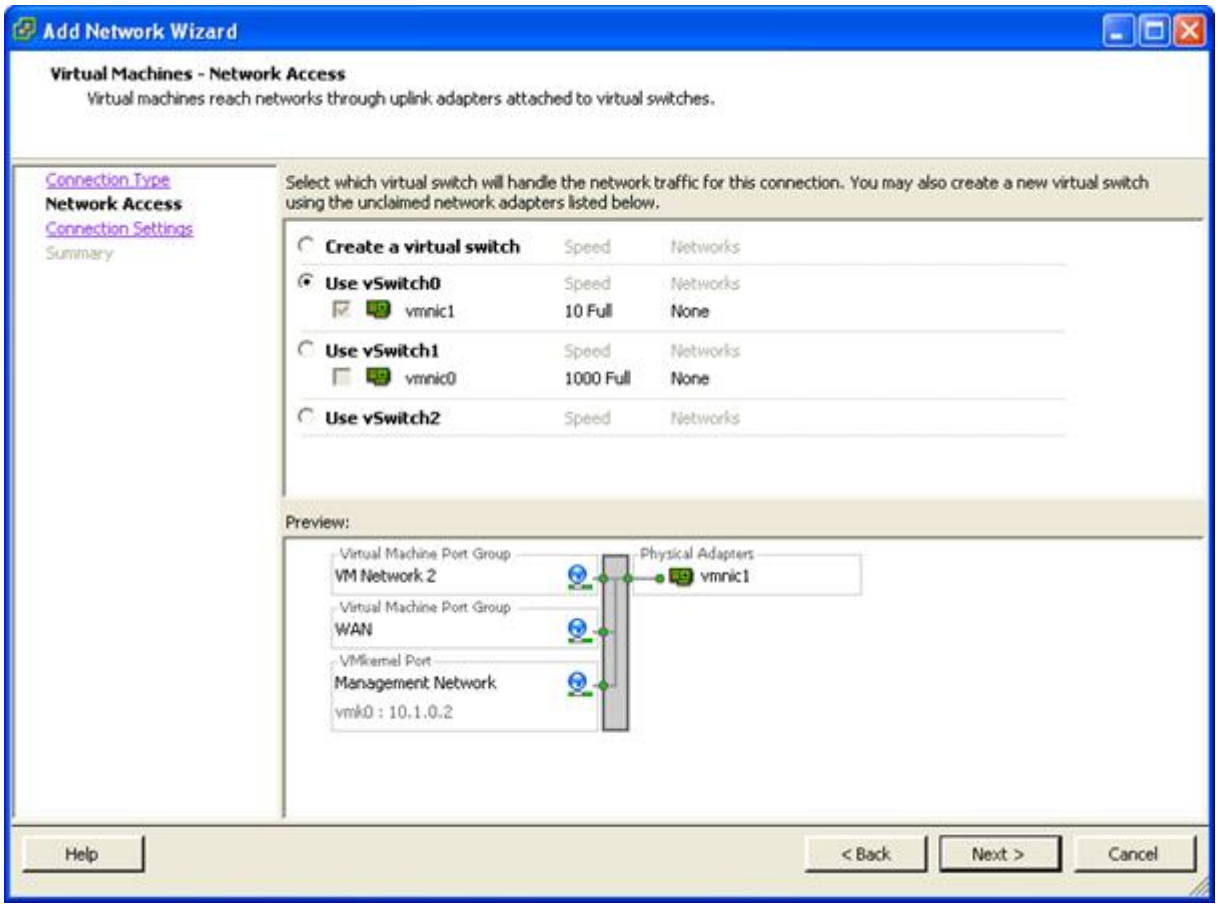
Navidezno omrežje znotraj strežnika je bila prva funkcionalnost, ki smo jo vzpostavili (Slika 8).



Slika 8: Načrtovanje navideznega omrežja

Za naš strežnik smo izbrali računalnik Dell OptiPlex 755 z dvema mrežnima vmesnikoma. Kot rečeno, smo uporabili le enega, ki je povezan direktno v internet, drugi pa nam je na voljo, če bi omrežje želeli širiti še na več fizičnih delovnih postaj. Znotraj strežnika ESXi smo ustvarili dve navidezni omrežji, ki sta med seboj ločeni. Čeprav gre za navidezna omrežja, le-ta simulirajo fizična, zato lahko tudi znotraj teh ustvarimo virtualna omrežja VLAN, vendar v našem primeru to ni bilo smiselno. Kot vmesni člen med obema omrežjema smo vzpostavili strežnik s sistemom Vyatta (poglavje 2.4.1.), ki služi kot požarni zid in mrežni usmerjevalnik. Šele znotraj drugega omrežja pa se nahajajo aplikativni in bazni strežniki, na katerih dejansko gostujejo storitve in podatki.

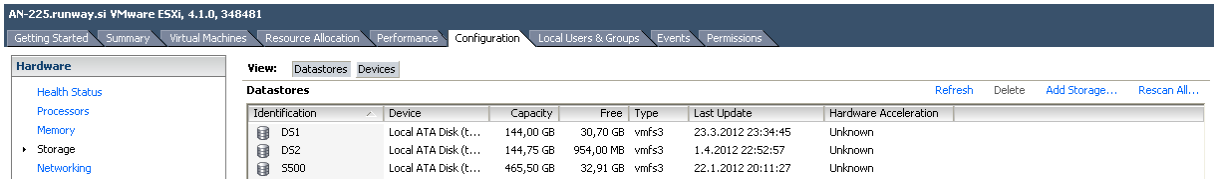
Ustvariti smo morali dve navidezni mrežni stikali, eno s povezavo na fizični mrežni vmesnik in drugo, ki je brez njega. Vsak mrežni vmesnik lahko namreč uporabimo le za eno virtualno mrežno stikalo. Stikali smo ustvarili uporabo čarovnika za ustvarjanje virtualnih omrežij, ki nam je na voljo s klikom na **Networking** (Slika 7) in nato **Add Networking...**. Čarovnik je za uporabo preprost, izbrati smo morali tip omrežja in po potrebi izbrali fizične mrežne vmesnike, na katere naj bo novo stikalo povezano (Slika 9).



Slika 9: Ustvarjanje navideznega omrežja

Drug pomemben vidik pri načrtovanju je upravljanje s pomnilniškimi kapacitetami. Ustvariti smo morali podatkovna polja (ang. datastore), ter jih kasneje uporabili za navidezne trde diske pri kreiranju virtualnih strežnikov.

Podatkovna polja je možno ustvariti iz fizičnih diskov v strežniku, iz omrežnega diskovnega polja ali iz predhodno ustvarjenega polja. Naš strežnik je imel vgrajene tri fizične trde diske SATA in iz vsakega od teh smo ustvarili svoje podatkovno polje (Slika 10). Polja ustvarimo s klikom na **Add storage**, ki se nahaja pod **Storage** na zavihku **Configuration**, seznam fizičnih naprav za hranjenje podatkov pa nam je na voljo ob kliku na gumb **Devices**.

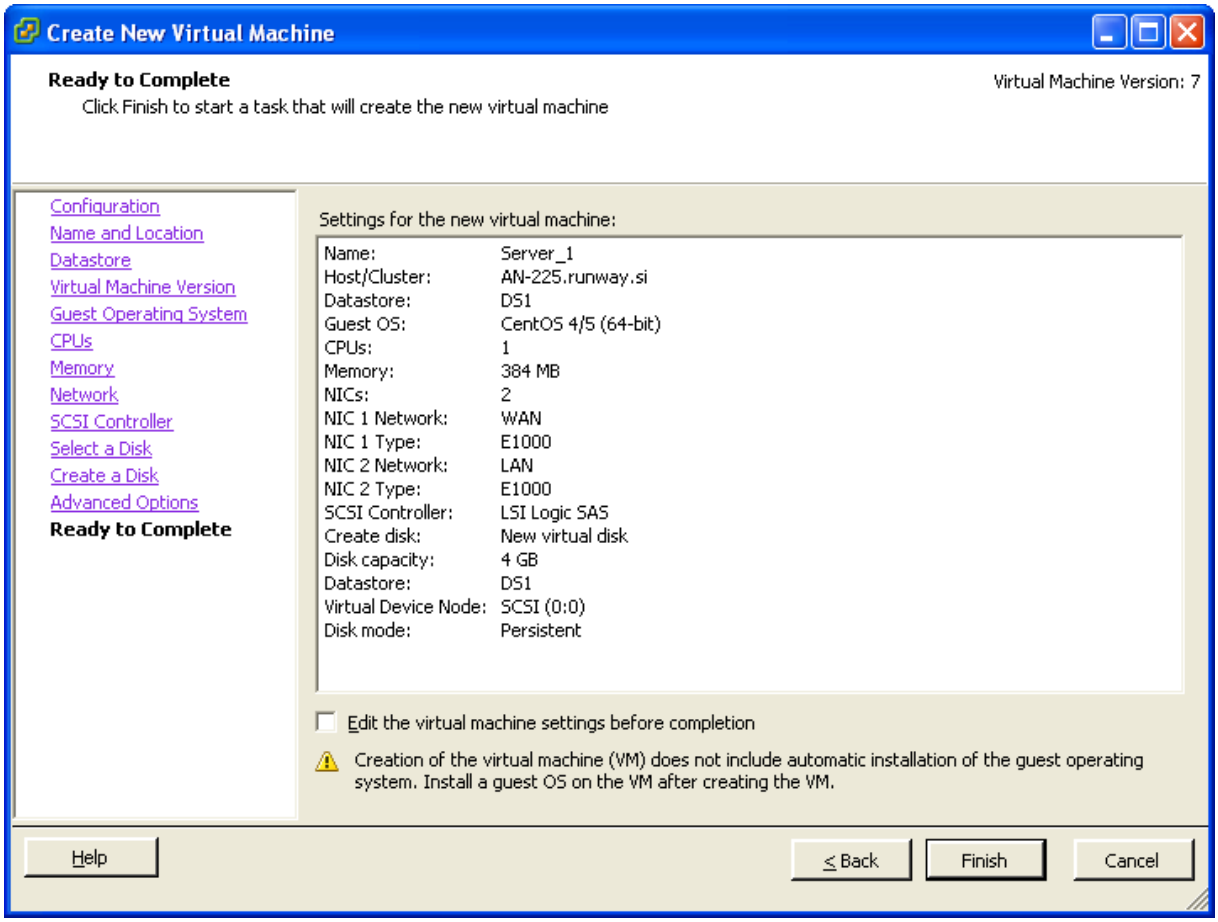


Slika 10: Pregled ustvarjenih podatkovnih polj

2.4. Postavitev sistemov

Po ustvaritvi slike omrežja in imetju ustreznih pomnilniških kapacitet smo lahko pričeli z ustvarjanjem virtualnih strežnikov. Uporabili smo čarovnika v aplikaciji, ki se zažene, ko izberemo **Create a new virtual machine** pod zavihkom **Getting Started**.

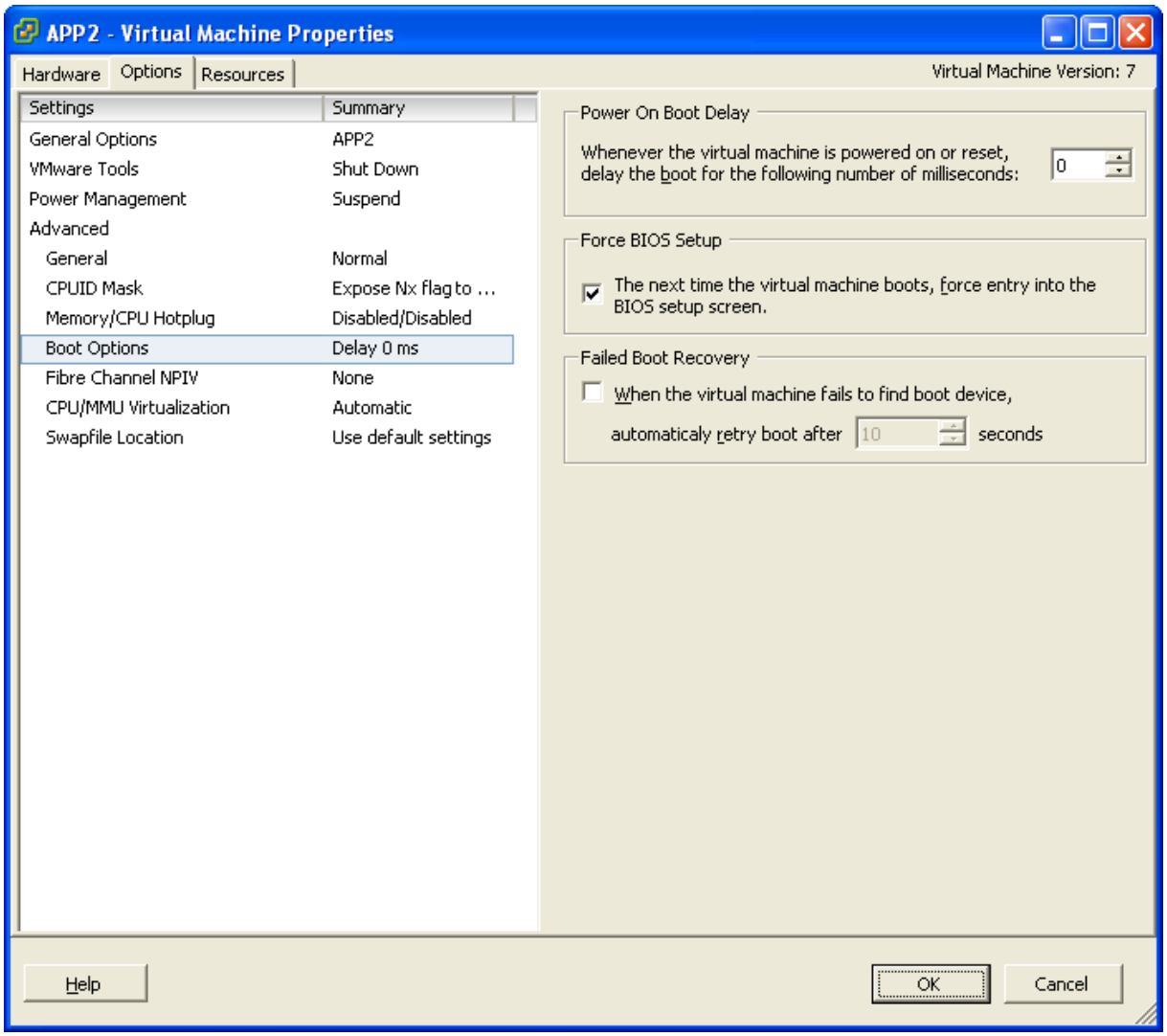
S čarovnikom je ustvarjanje virtualnega računalnika preprosto, saj nas po korakih vodi čez posamezne stopnje. V prvem koraku smo izbrali konfiguracijo po meri, saj se s tem izognemo morebitnemu spreminjanju nastavitev kasneje. Nadalje smo virtualni strežnik poljubno poimenovali, kjer priporočamo, da se držimo nekega sistema. Predvsem moramo paziti, če imamo na enem strežniku ESXi produkcijske in testne strežnike, pri slednjih je smiselno dodati "-TEST" oz. drugo ustrezno oznako, ki nedvoumno določa, da gre za testni sistem. Nadaljnji koraki so precej preprosti, izbrali smo podatkovno polje ter operacijski sistem, ki smo ga namestili na strežnik ter mu dodelili navidezno strojno opremo. Postopek smo zaključili s preverbo nastavitev in klikom na gumb **Finish** (Slika 11).



Slika 11: Pregled nastavitev virtualnega strežnika

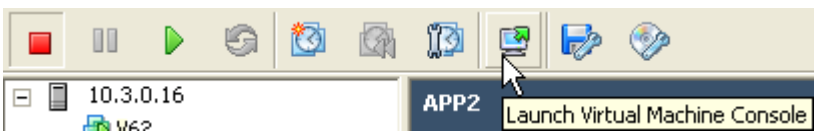
Virtualni strežnik je sedaj viden v aplikaciji, od koder ga imamo možnost zagnati, izbrisati in tudi spremeniti večino nastavitev. Vse te možnosti so nam na voljo s klikom na **Edit virtual machine**

settings, kjer lahko upravljamo z vsemi nastavitvami, ki jih aplikacija omogoča. Kot prvo smo morali zahtevati vstop v sistem BIOS ob zagonu. To smo storili na zavihku **Options** (Slika 12), ki se nahaja med **Hardware** (ang. strojna oprema) in **Resources** (ang. sredstva).



Slika 12: Upravljanje z nastavitvami virtualnega strežnika

Na strežnik se še ne moremo povezati z uporabo aplikacije, ki omogoča oddaljeni dostop, saj na strežniku še ni nameščenega operacijskega sistema, posledično pa tudi nobena druga aplikacija. vSphere Client nam zato ponuja možnost uporabe konzolnega okna. Le-ta simulira sliko na monitorju, ki bi jo sicer videli pri fizičnem strežniku. Okno odpremo s klikom na **Launch Virtual Machine Console** v menijski vrstici (Slika 13).



Slika 13: Zagon konzolnega okna

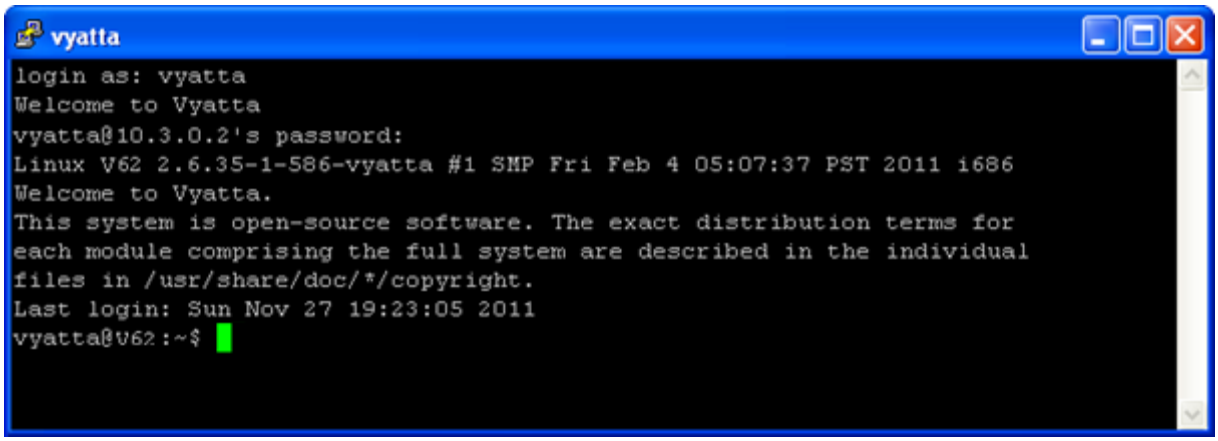
Kot smo zahtevali, bo sistem ob zagonu samodejno vstopil v sistem BIOS. Ker je strežnik na tej točki samo navidezni stroj smo morali nastaviti primarni zagonski medij, ki bo v našem primeru datoteka .ISO, ki simulira namestitveni medij CD. Nastavitev se je uveljavila ob ponovnem zagonu, ki je sledil shranjevanju sprememb in izhodu iz sistema BIOS. Nadaljnji postopek se od operacijskega sistema na fizičnem računalniku ne razlikuje.

2.4.1. Strežnik Vyatta 6.2

Vyatta je sistem, ki omogoča funkcionalnosti požarnega zidu, usmerjanja prometa, ustvarjanje tunelov VPN, zaznavanja in preprečevanja vdorov, ter še mnogo drugih. Gre za posebno distribucijo sistema Linux, ki jo namestimo enako kot druge operacijske sisteme, lahko pa tudi prenesemo[4] že sliko virtualnega strežnika, ki jo zgolj uvozimo v strežnik ESXi.

Za vzpostavitev strežnika, ki bo gostil sistem Vyatta smo se ravnali po navodilih, opisanih v poglavju 2.4, pri čemer smo strežniku dodali dve virtualni mrežni kartici od katerih je vsaka povezana na svoje virtualno mrežno stikalo. Nadalje smo nanj namestili sistem Vyatta, ki ga je bilo potrebno še ustrezno konfigurirati. Sistem bomo primarno uporabljali kot požarni zid in mrežni usmerjevalnik za kar smo morali nastaviti vrata (ang. port), preko katerih je naš notranji sistem dostopen ter ustrezne prehode, preko katerih bomo dostopali do drugih omrežij, predvsem interneta.

Po namestitvi smo se že lahko prijavili v sistem in z ukazom `set service ssh` tudi omogočili oddaljeni dostop preko protokola SSH (Slika 14). Vse funkcionalnosti, ki jih sistem omogoča lahko nastavimo z vnosom ukazov preko konzole, vendar je bolj smiseln spletni vmesnik, ki smo ga omogočili z ukazom `set service https`. Vse nastavitve je potrebno tudi uveljaviti z ukazom `commit`.



Slika 14: Prijava na strežnik Vyatta

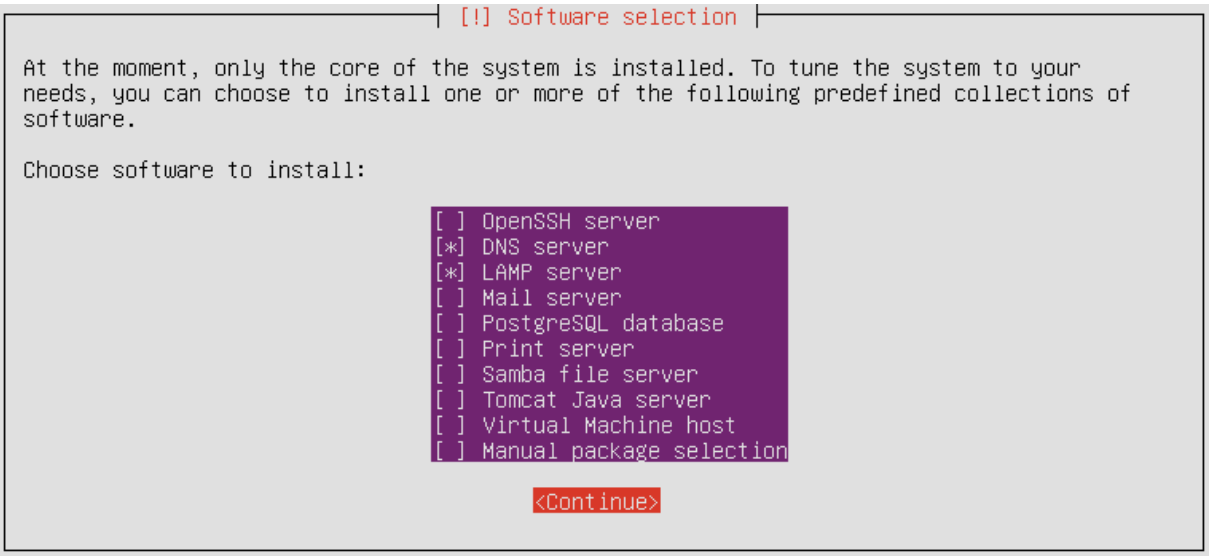
[4] https://solutionexchange.vmware.com/store

V veliko pomoč pri konfiguraciji sistema Vyatta nam je tudi obsežna dokumentacija [8], ki jasno opisuje vse nastavitve, ki jih sistem ponuja. Nastavitve v našem sistemu smo opisali v poglavjih 2.5.2., 2.5.3. in 2.5.4.

2.4.2. Strežnik Ubuntu Server 11.04

Sistem Ubuntu je trenutno[5] eden najbolj priljubljenih brezplačnih sistemov za domačo uporabo, zato smo se odločili, da strežniško verzijo sistema (Ubuntu Server) uporabimo tudi pri našem projektu.

Priprava virtualnega okolja za gostovanje strežnika je bila enaka kot pri sistemu Vyatta, sama namestitev se tudi ne razlikuje bistveno. Izbrati smo morali nekaj splošnih nastavitev kot so jezik, datotečni sistem in podobno, pomembnejša pa je izbira med programsko opremo, ki jo lahko hkrati namestimo na strežnik. Na voljo nam je več možnosti (Slika 15), izbrali pa smo le strežnik DNS, ki skrbi za sistem domenskih imen ter LAMP. Slednji označuje komplet programske opreme, ki se namesti na strežnik **L**inux in vključuje spletni strežnik **A**pache, podatkovno bazo **M**ySQL in podporo za jezik **P**HP.



Slika 15: Izbira dodatne programske opreme

Kasneje smo postavili še dva taka strežnika, pri enemu smo namestili poštni strežnik (Mail Server), pri drugemu pa ničesar, saj smo nanj dodatno namestili podatkovno bazo MySQL. Le-ta namreč zgolj hrani kopijo baze, za katero skrbi prvi strežnik.

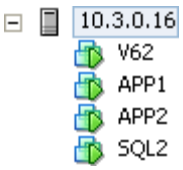
Sistem Ubuntu Server se privzeto namesti brez grafičnega uporabniškega vmesnika, saj ga strežnik za opravljanje svojih storitev tudi ne potrebuje. Ta pristop zmanjšuje potrebo po sistemskih sredstvih ter zvišuje nivo varnosti, saj sistema ni potrebno posodabljati z varnostnimi popravki za

[5] Leta 2012

storitve, ki jih zahteva ali so vključene v grafični uporabniški vmesnik. Tovrstno prakso je pričel povzemati tudi Microsoft z operacijskim sistemom Windows 2008 Core.

Grafični uporabniški vmesnik vseeno lahko dodatno namestimo, na voljo imamo celo več možnosti. Sistem Ubuntu namreč ponuja več namizij, zanimiv pa je tudi spletni uporabniški vmesnik, ki je sicer manj zmogljiv, vendar zahteva tudi najmanj sistemskih sredstev.

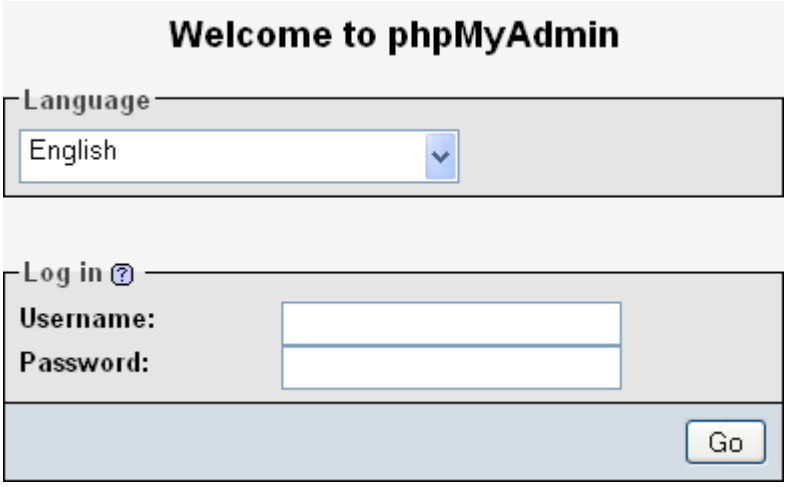
Po končanih namestitvah imamo tako na voljo 4 delujoče virtualne strežnike, kot smo tudi načrtovali (Sliki 8 in 16).



Slika 16: Pregled virtualnih strežnikov v aplikaciji vSphere Client

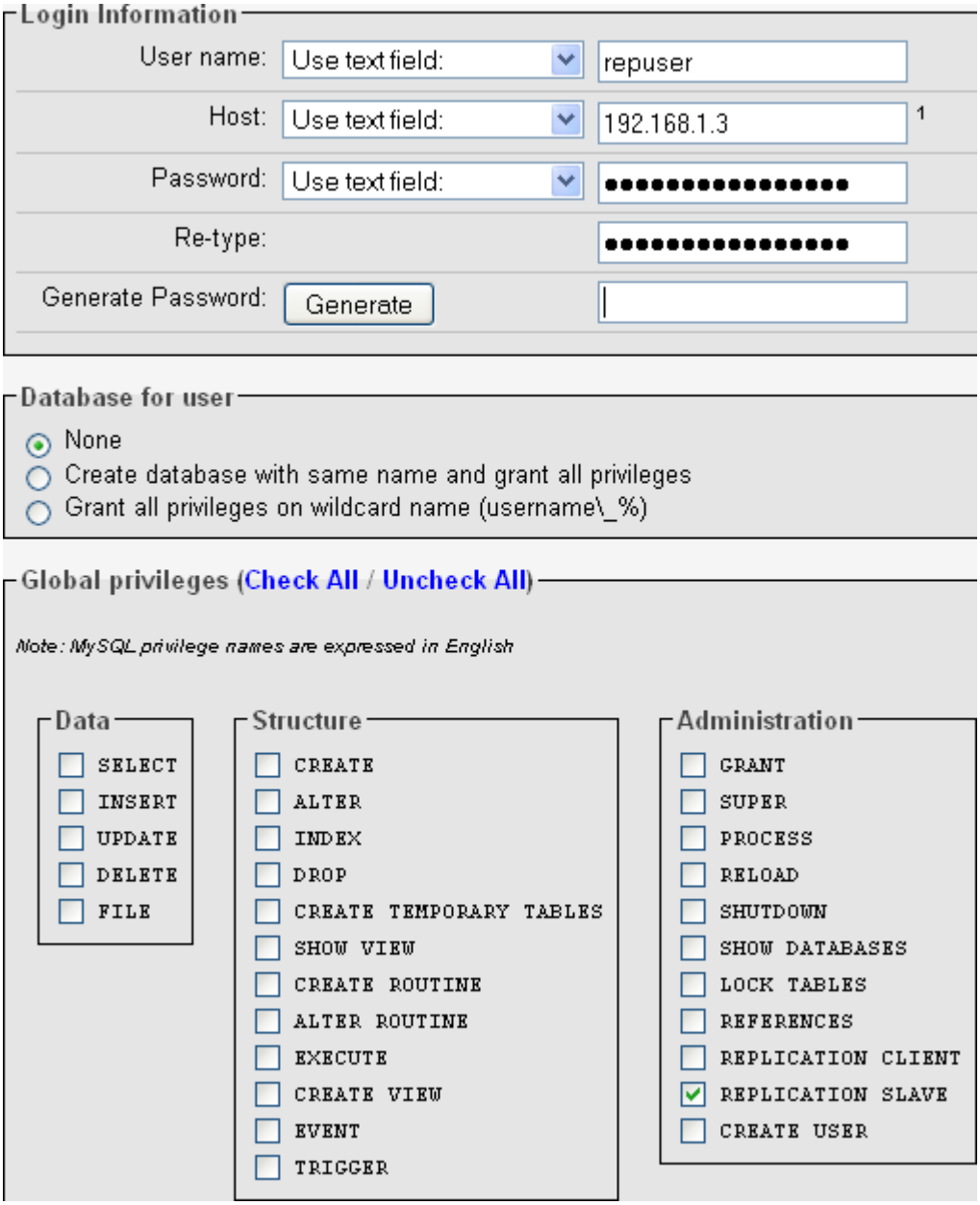
2.4.3. Podatkovna baza MySQL 5.1

Zadnji Ubuntu strežnik (SQL2) bo, kot smo že omenili, gostil le kopijo podatkovne baze MySQL, ki se nahaja na prvem strežniku (APP1). Ker baze nismo namestili hkrati z operacijskim sistemom, smo to morali storiti naknadno. Sam postopek namestitve je preprost, po prijavi na strežnik v ukazno vrstico vpišemo `sudo apt-get install mysql-server`. Po podatkovni bazi smo namestili tudi spletni grafični uporabniški vmesnik, ki močno olajša delo s podatkovno bazi in sicer z ukazom `sudo_apt-get_install_phpmyadmin`. Po uspešni namestitvi v spletni brskalnik vpišemo naslov http://naslov.ip/phpmyadmin (kje je "naslov.ip" naslov IP našega strežnika) in odpre se nam spletna stran, ki od nas zahteva uporabniško ime in geslo (Slika 17). Spletni vmesnik smo zaradi lažje administracije namestili na oba naša strežnika, na katerem teče baza MySQL.



Slika 17: Spletni uporabniški vmesnik za podatkovno bazo MySQL

Preko tega vmesnika lahko postorimo vse, kar nam podatkovna baza omogoča. V našem primeru smo nastavili replikacijo med bazama na strežnikih APP1 in SQL2. V prvem koraku smo morali na primarni (ang. master) bazi ustvariti uporabnika z ustreznimi pravicami, ki ga sekundarni (ang. slave) strežnik uporablja za dostop. Baza MySQL ima za takega uporabnika določen poseben nivo pravic **Replication Slave** (Slika 18). Priporočljivo je tudi nastaviti naslov IP strežnika, od koder bodo prihajale zahteve po podatkih z uporabo tega uporabnika (polje **Host**). Tako zmanjšamo tveganje neupravičenega dostopa do podatkov, saj je le-ta omejen tudi na izvorni naslov IP od koder prihaja zahteva.



Slika 18: Ustvarjanje uporabnika za replikacijo baze

Ko smo podatke o naslovu IP primarnega strežnika in poverilnice uporabnika za replikacijo vpisali v sekundarni strežnik (funkcija **Slave configuration**), se je replikacija pričela izvajati na vseh podatkovnih bazah.

2.5. Nastavitev sistema

Ko smo postavili vse strežnike, smo se lotili ključnih sistemskih nastavitev, s katerimi smo omogočili povezljivost sistemov ter zvišali varnost pred zunanjimi napadi. Nastavitve so podrobneje opisane v naslednjih poglavjih.

2.5.1. Sistem domenskih imen

Sistem domenskih domen upravlja storitev DNS, ki je teče na strežniku APP1. Ta razrešuje domenske naslove v naslove IP v dveh smereh:

- rekurzivno: ko kateri izmed strežnikov zahteva dostop do podatkov po domenskem naslovu, strežnik DNS povpraša druge strežnike DNS po ustreznem naslovu IP, ter ga sporoči strežniku, ki je zahteval dostop.
- avtoritativno: ko želi kak strežnik ali odjemalec dostopati do storitev na naših strežnikih po domenskem naslovu, mu naš strežnik DNS posreduje ustrezen naslov IP.

Da bo naš strežnik lahko ustrezno sporočal naslove IP smo morali zadostiti več zahtevam;

- registrarju domene sporočiti naslov IP našega strežnika NS (ang. name server), ki je kar naš strežnik APP1,
- v nastavitve vpisati naslove IP zunanjih strežnikov DNS,
- urediti translacijo NAT in sprostiti vrata 53 do našega strežnika DNS in
- ustrezno konfigurirati ter v strežnik vnesti ustrezne zapise.

Programsko opremo za delovanje storitve DNS smo že namestili hkrati z operacijskim sistemom, zato smo lahko takoj prešli na samo konfiguracijo. To smo storili v petih različnih datotekah na strežniku APP1, kakor je opisano v nadaljevanju.

V datoteki `/etc/bind/named.conf.local` smo kreirali svojo domeno na strežniku DNS, ki je enaka domeni, ki smo jo registrirali pri našem registrarju spletnih domen. Če bi domeno uporabljali zgolj znotraj lastnega omrežja (intranet), pa bi lahko uporabili poljubno ime domene, ki pa mora biti skladno s RFC 1035 [14]. Vsebina datoteke je naslednja:

```
zone "domena.si" {
type master;
file "/etc/bind/zones/domena.si.db";
};
zone "3.0.10.in-addr.arpa" {
type master;
file "/etc/bind/zones/rev.3.0.10.in-addr.arpa";
};
```

Kreirali smo novo cono za domeno domena.si ter ustrezno polje naslovov IP, ki jih bodo uporabljale naprave v tej domeni.

V datoteki `/etc/bind/named.conf.local` smo vpisali naslove IP strežnikov DNS, da bo lahko naš strežnik DNS razreševal tudi tuje domenske naslove:

```
forwarders {
    84.255.209.79;
    84.255.210.79;
    193.2.1.66;
    8.8.8.8;
};
```

Vnesli smo štiri strežnike DNS, prva dva sta od našega ponudnika interneta, tretji je od zavoda ARNES, ki skrbi za vse .si domene, zadnji pa Googlov javni strežni DNS. Vpis tolikih strežnikov sicer ni nujen, priporočljiva pa sta vsaj dva.

Vnesti smo morali tudi naslov IP našega strežnika v datoteko `/etc/resolv.conf`:

```
search domena.si.
nameserver 10.3.0.10
```

Sedaj je bilo potrebno še definirati cono, ki smo jo kreirali nekoliko prej. To storimo v datoteki `/etc/bind/zones/domena.si.db`.

```
$TTL 1H
@ IN SOA app1.domena.si. webmaster.domena.si. (
   2012053001
   7200
   3600
   604800
   3600
);
domena.si.      IN      NS          app1.domena.si.
app1            IN      A           10.3.0.10
app2            IN      A           10.3.0.11
www             IN      CNAME       app2
mail            IN      CNAME       app2
                IN      MX          10 mail.domena.si
```

V prvi vrstici smo nastavili TTL (ang. Time to Live), ki določa čas veljavnosti zapisa v sekundah. Strežniki DNS namreč ne poizvedujejo po naslovu IP vsakič, ko kdo zahteva po njem, temveč jih shranijo za čas TTL. Zapisi DNS se namreč ne spreminjajo pogosto in ta način močno razbremenjuje strežnike.

Druga vrstica se začne z "IN", kar je oznaka za internet. Druge oznake se sicer ne uporabljajo več, vendar je ta še vedno potrebna. Sledi ime SOA (ang. State of Art) strežnika, ki je avtoritativen za domeno, kar je pri nas strežnik APP1. Temu je dodan e-poštni naslov administratorja, kjer pa se namesto znaka '@' uporablja pika. Obema zapisoma mora slediti pika tudi na koncu, vrstica pa se mora končati z oklepajem.

Nadaljnje vrstice vsebujejo številčne vrednosti, prva je serijska številka, ki se spremeni ob vsaki spremembi cone. Drugi strežniki DNS namreč tudi po preteku TTL ne bodo prenesli vseh

podatkov, če se številka ni spremenila. Ostali zapisi so časovne vrednosti v sekundah za osvežitev, ponovni poizkus, potek in minimalni TTL (ang. refresh, retry, expire in minimum TTL). Tu smo vnesli splošno priporočljive vrednosti. Sledi zaklepaj in podpičje v novi vrstici.

V naslednjih vrsticah so dejanski zapisi DNS, ki domenska imena povezujejo z naslovi IP. Zapisi so določenega tipa, ki označuje njihovo uporabo. V naš strežnik DNS smo vpisali dva zapisa A, ki za strežnika APP1 in APP2 hranita ustrezna naslova IP, ter dva aliasa (zapis CNAME), ki usmerjata domenski naslov na nek drug zapis. To je uporabno iz dveh vidikov; en strežnik je lahko dosegljiv preko več imen, ter lažje ga je zamenjati z drugim strežnikom, ne da bi bila strežniška storitev nedosegljiva. Dodatni zapisi A se bodo dodali samodejno, ko se bo katera izmed naprav prijavila v domeno. Pomemben je tudi zapis NS, ki označuje strežnik DNS, ki hrani zapise za domeno in je v našem primeru kar strežnik APP1. Zapis MX pa označuje strežnik, ki za domeno sprejema elektronsko pošto. Teh zapisov je lahko tudi več, vsak pa ima tudi prioriteto, označeno s številko pred imenom strežnika. Nižja številka pomeni višjo prioriteto, strežnikov z enako prioriteto pa je lahko več.

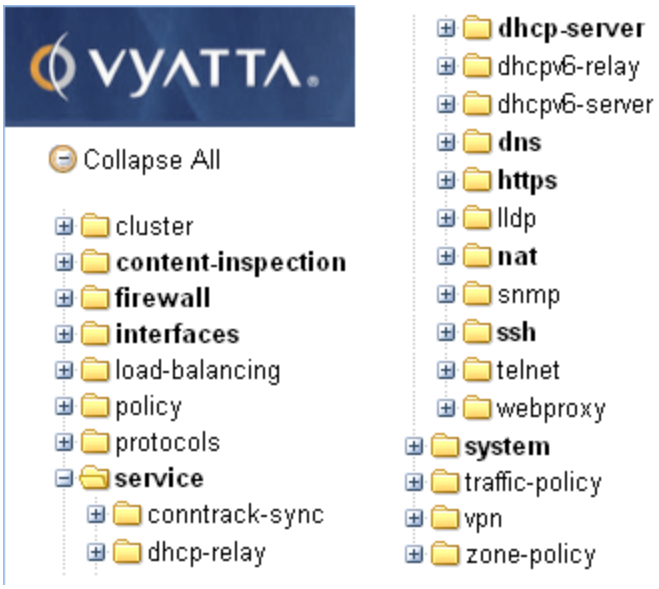
Zadnja potrebna datoteka je `/etc/bind/zones/rev.0.3.10.in-addr.arpa`. S pomočjo te datoteke izvajamo vzvratne poizvedbe, torej iz naslovov IP v domenska imena. Struktura je podobna kot pri datoteki `domena.si.db`.

```
$TTL 1H
@       IN      SOA     ns.domena.si. webmaster.domena.si. (
                        2012053001
                        7200
                        3600
                        604800
                        3600
)
        IN      NS      ns.domena.si.
10      IN      PTR     app1.domena.si.
11      IN      PTR     app2.domena.si.
```

Strežnik DNS sedaj ponovno zaženemo z ukazom `sudo /etc/init.d/bind9 restart`, kar sproži uveljavitev sprememb in s tem delovanje v polni funkcionalnosti.

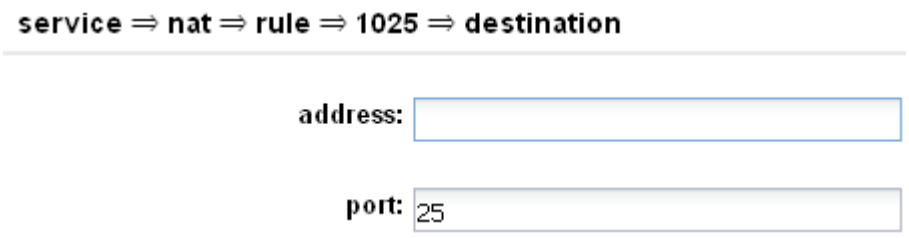
2.5.2. Usmerjanje prometa

Ker naši strežniki, ki gostujejo storitve, niso neposredno povezani v internet, smo morali urediti povezljivost do njih. Vmesni člen med internetom in internim omrežjem je strežnik s sistemom Vyatta, ki omogoča tovrstno usmerjanje prometa. Upravljanje s sistemom smo izvedli preko spletnega vmesnika (poglavje 2.4.1.). Sistem nam ponuja veliko storitev, med drugim tudi požarni zid (poglavje 2.5.3.), strežnik DHCP in DNS, ter druge (Slika 19). Usmerjanje prometa nastavljamo pri razdelku **nat** (ang. Network address translation).

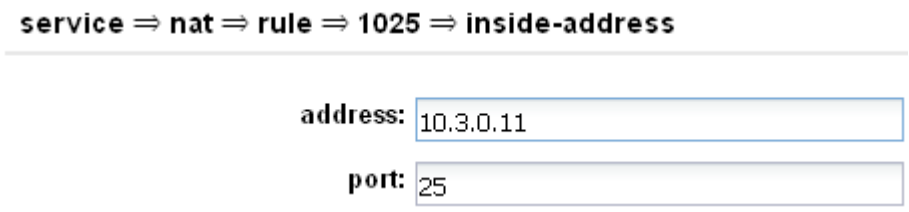


Slika 19: Storitve, ki jih ponuja sistem Vyatta

Vsako preusmeritev prometa moramo izvesti z nekim pravilom (ang. rule), ki je številčno poimenovano in ima svoj obseg parametrov. Obvezna sta le `destination` in `inside-address`, ki določata kam se usmerja promet, ki je naslovljen na določena vrata. Sliki 20 in 21 prikazujeta kako smo uredili preusmerjanje prometa skozi vrata 25, ki jih uporablja naš poštni strežnik.



Slika 20: Nastavitev parametra 'destination'



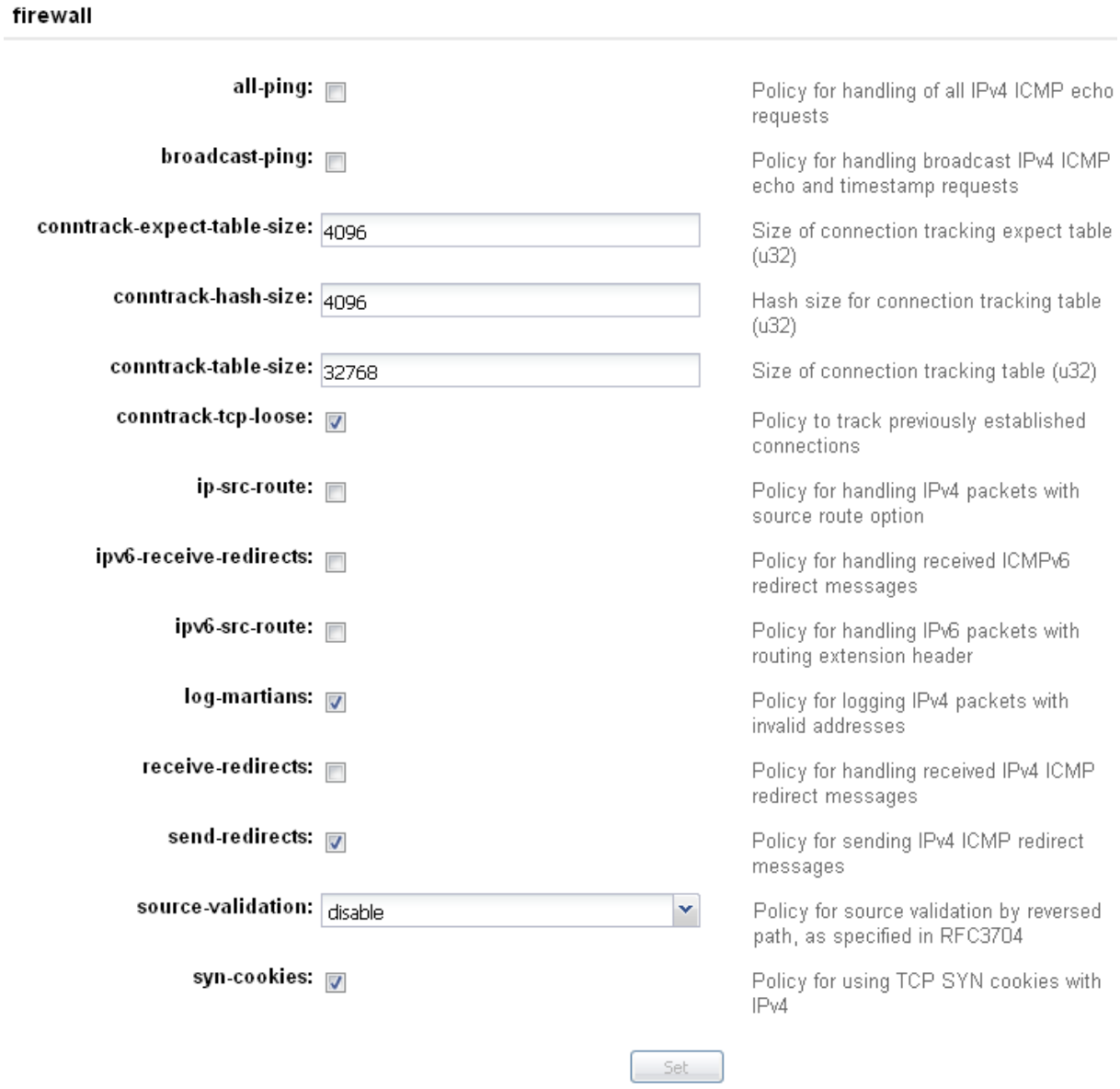
Slika 21: Nastavitev parametra 'inside-address'

Nastavitve smo uveljavili s klikom na gumb **Commit**, vendar se le-te izgubijo ob ponovnem zagonu strežnika, če konfiguracije ne shranimo (gumb **Save**). Strežnik se sedaj odziva in odgovarja na zahteve, ki ji prožimo preko vrat 25 tudi izven notranjega omrežja.

2.5.3. Požarni zid

Ena izmed storitev sistema Vyatta je tudi požarni zid, s katerim bistveno pripomoremo k varnosti sistema. Požarni zid namreč blokira podatkovni promet, ki ga na podlagi določenih kriterijev zazna kot potencialno nevaren. Nekatere kriterije nastavimo globalno za celoten požarni zid (Slika 22), za ostale pa smo ustvarili skupino pravil (ang. rule), ki ima svoje ime, opis (neobvezno) in privzeto akcijo. Možne vrednosti privzete akcije so:

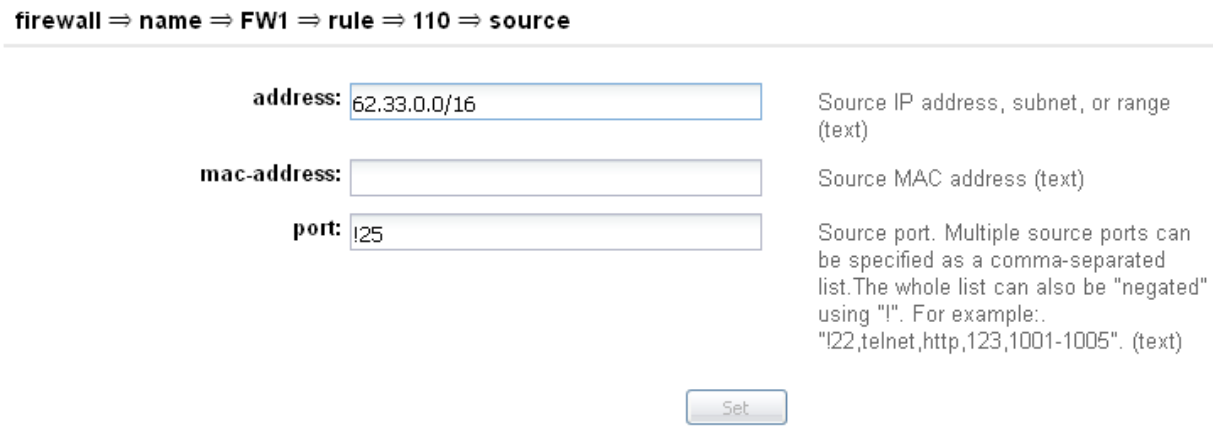
- **accept**; promet se smatra kot varen in je sproščen,
- **drop**; promet se smatra kot nevaren in se zavrže in
- **reject**; promet se smatra kot nevaren in se zavrže, pošiljatelju je poslan paket TCP reset.



Slika 22: Globalne nastavitve požarnega zidu

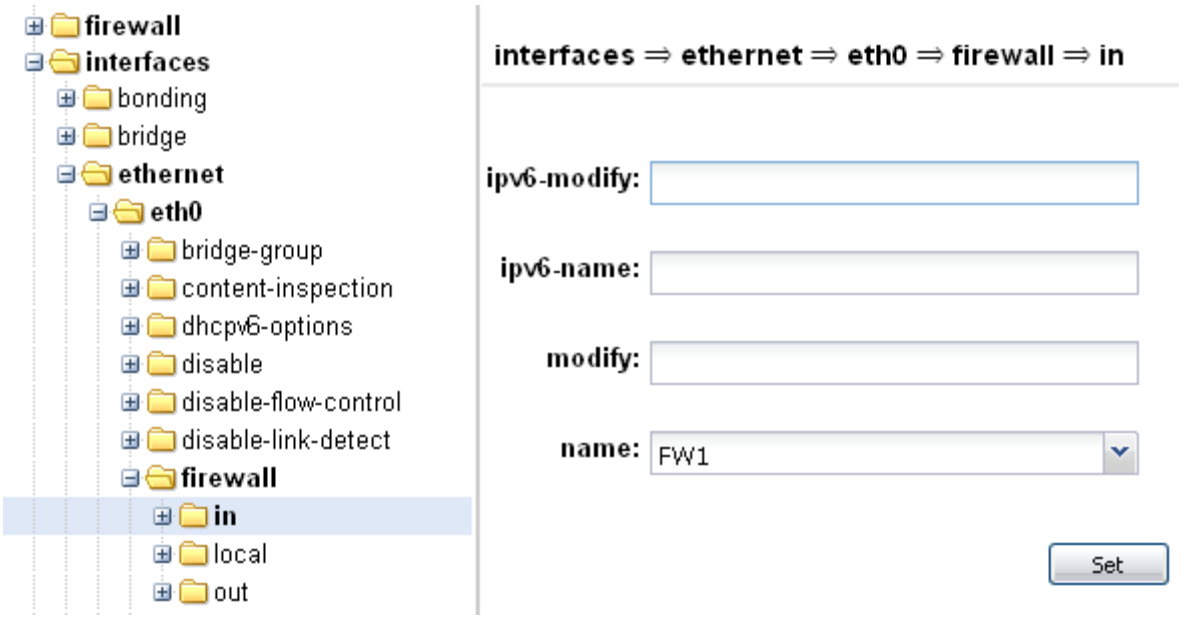
Našo skupino smo poimenovali FW1 ter v njej ustvarili eno pravilo. Pravila so številčno poimenovana, imajo opis (neobvezno), shranjujejo dnevniški zapis (izbirno) imajo določen protokol (neobvezno) in seveda akcijo kaj storiti s prometom, če pravilu ustreza. Ob treh akcijah, ki smo jih našteli v prejšnjem odstavku, imamo še možnost **inspect**, ki promet pošlje sistemu za zaznavo vdorov, ki ga dodatno preveri ter na podlagi lastnih kriterijev zavrne ali sprosti (poglavje 2.5.4.). Poimenovanje pravil je pomembno, saj se promet preverja zaporedno po vseh pravilih, zato je lahko nek promet zavrnjen tudi, če je ustrezalo parametrom katerega od prejšnjih pravil z akcijo **accept**. Če promet ne ustreza nobenemu izmed pravil se zanj uveljavi privzeta akcija. Smiselno je zato pravilo, kjer pričakujemo, da bo ustrezalo največji količini prometa poimenovati z najnižjo številko.

Pravilu lahko določimo veliko parametrov, od osnovnih, kot so tip protokola, pa do časovnih omejitev, kdaj naj pravilo deluje. Vsa so opisana v dokumentu Vyatta Firewall Reference Guide [9], pri našem pravilu, ki promet zavrže, pa smo se omejili le na izvor, ki ga definiramo v razdelku **source**. Tu smo nastavili, da promet ustreza pravilu, če prihaja iz naslova IP v rangu `62.33.0.0/16` (polje **address**). Gre za segment naslovov IP, ki so dodeljeni uporabnikom interneta v Rusiji, za katere ni smiselno pričakovati, da bodo uporabljali storitve na naših strežnikih. Ker pa nam bodo morda vseeno želeli poslati elektronsko pošto, smo dovolili uporabo vrat 25 z vpisom `!25` (Slika 23). Možno je tudi nastaviti izvorni naslov MAC, kar pa v tem primeru ni bilo smiselno.



Slika 23: Nastavitev 'source' pri pravilu požarnega zidu

Da je pravilo tudi zares delovalo smo morali skupino FW1 uveljaviti še na mrežnem vmesniku. Ker naše pravilo filtrira promet, ki prihaja iz smeri interneta v naše notranje omrežje, smo skupino uveljavili na vmesniku **eth0** za promet **in** (Slika 24). Nato smo morali le še uveljaviti spremembe na sistemu Vyatta.



Slika 24: Uveljavitev skupine pravil na mrežni vmesnik

2.5.4. Sistem za preprečevanje vdorov

Vyatta vsebuje tudi osnoven sistem za preprečevanje vdorov (ang. Intrusion Prevention System – IPS). Ta sistem spremlja promet in zaznava izvajanje potencialno zlonamerne kode, ter jo zaustavi v realnem času. V ta namen Vyatta uporablja sistem Snort, ki je eden izmed najboljših brezplačnih sistemov za zaznavo in preprečevanje vdorov [12].

Za uporabo sistema Snort smo se morali registrirati na spletni strani http://www.snort.org/, ter prejeli posebno kodo 'oinkcode', s katero sistemu Vyatta omogočimo redno posodabljanje namenskih pravil, na podlagi katerih se promet določi kot varen ali nevaren. Vsakodnevno se namreč srečujemo z novimi načini napada in posodabljanje vsake programske opreme je ključnega pomena. Vnos kode se izvede z ukazom set, ki smo ga preko spletnega vmesnika izvedli preko zavihka **Operation**. Od tam smo razprli drevesno strukturo na ustreznem mestu ter vpisali kodo (Slika 25).

Slika 25: Nastavitev kode 'oinkcode'

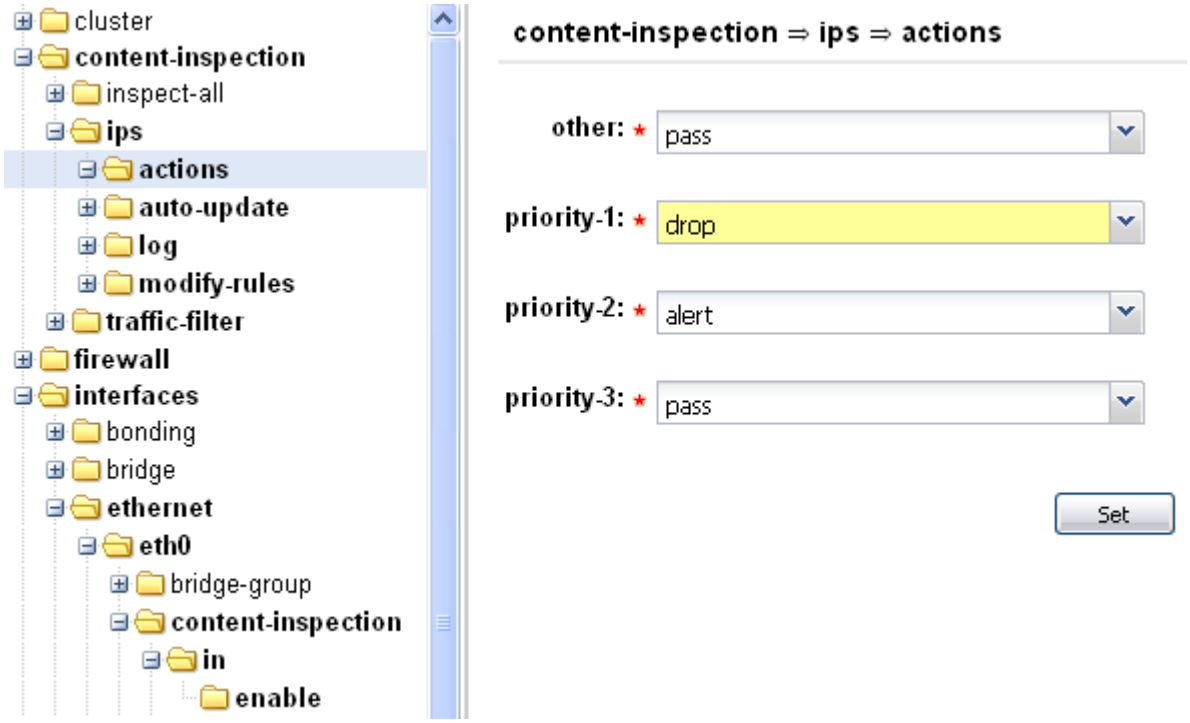
Snort hrani dnevniške zapise za sumljiv promet, če tako nastavimo, za kar potrebuje dostop do podatkovne baze. Podatkovni strežnik smo že vzpostavili zato smo se lotili postavitve namenske podatkovne baze z ustrezno strukturo ter kreiranje uporabniškega dostopa do baze. Strukturo baze smo pridobili v namestitvenem paketu sistema Snort [11] (datoteka `create_mysql`), za uporabnika pa smo uporabili kar 'root'. Pri vzpostavitvi je namreč smiselno delati z uporabnikom, ki ima zagotovo dovolj pravic in kasneje kreirati namenskega, ki nepotrebnih pravic nima. S tem se izognemo morebitnemu nepotrebnemu iskanju napake nezmožnosti dostopanja do baze.

Za redno posodabljanje pravil smo se morali ponovno prestaviti na zavihek **Configuration**, kjer smo določili tudi, kje (**content-inspection→ips→modify-rules→internal-network**) ter kakšen promet naj se preverja (**interfaces→ethernet→eth0→content-inspection**). Naš sistem preverja promet znotraj našega internega omrežja, ki prihaja od zunaj. Možno je sicer tudi nastaviti preverjanje vsega prometa z možnostjo **inspect-all**.

Sedaj nam je preostala še nastavitev, kaj narediti s prometom, ki ga sistem zazna kot potencialno nevarnega. Sistem Snort zaznane sumljive aktivnosti loči v štiri prioritete, ki so obrazložene v dokumentu Vyatta System Security Reference Guide [10]. Vsaki od teh prioritet smo določili eno izmed štirih možnih akcij:

- **alert**; promet je sproščen, vendar se vnese zapis v dnevniško bazo,
- **drop**; promet se zavrže in v dnevniško bazo se vnese zapis,
- **pass**; promet se smatra kot varen in je sproščen in
- **sdrop**; promet se zavrže, zapis v dnevniško bazo pa se ne vnese.

Odločili smo se, da zavržemo promet v prioriteti 1, sprostimo, vendar hranimo zapis v bazi za promet prioritete 2, ter sprostimo preostali promet (Slika 26).



Slika 26: Nastavitev akcij ob sumljivem prometu

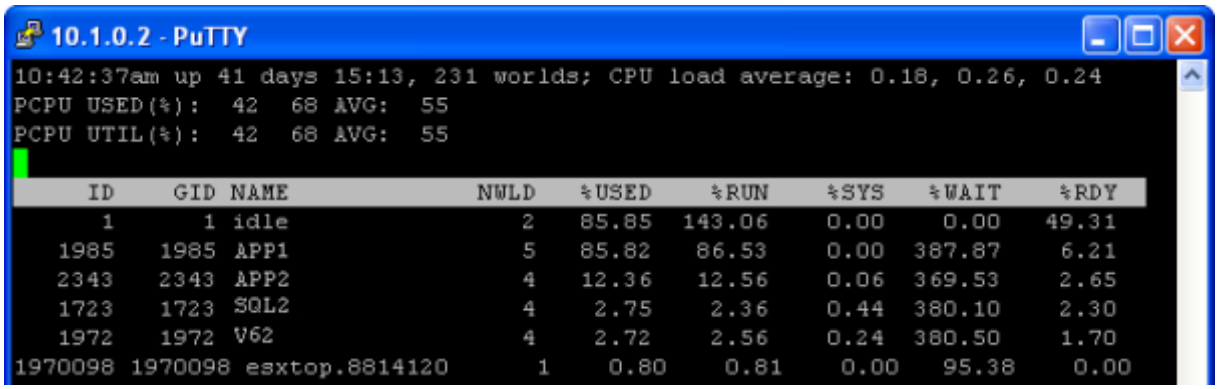
Za optimalno nastavitev sistema Snort je potrebno redno spremljanje dnevniških zapisov v podatkovni bazi ter ustrezno spreminjanje prioritet tipov prometa in akcij.

2.6. Testiranje

Ko smo sistem postavili po svojih željah in namenih, smo se lotili tudi testiranja osnovnih varnostnih in funkcionalnih nastavitev. Poiskali smo več različnih in brezplačnih spletnih aplikacij, ki opravijo osnoven zunanji pregled ter nas opozorijo na večje napake, ter tudi svetujejo kako jih odpraviti. Preverili smo tudi stabilnost in učinkovitost sistema samega.

2.6.1. Sistemski test

Pri pregledu učinkovitosti izrabe smo uporabili orodje esxtop, ki je že vključeno v programsko opremo ESXi. Uporaba je mogoča preko strežniške konzole ali protokola SSH, kot ukaz je potrebno le vpisati `esxtop` in prikaže se nam tabela s podatki o uporabi procesorske moči (Slika 27).

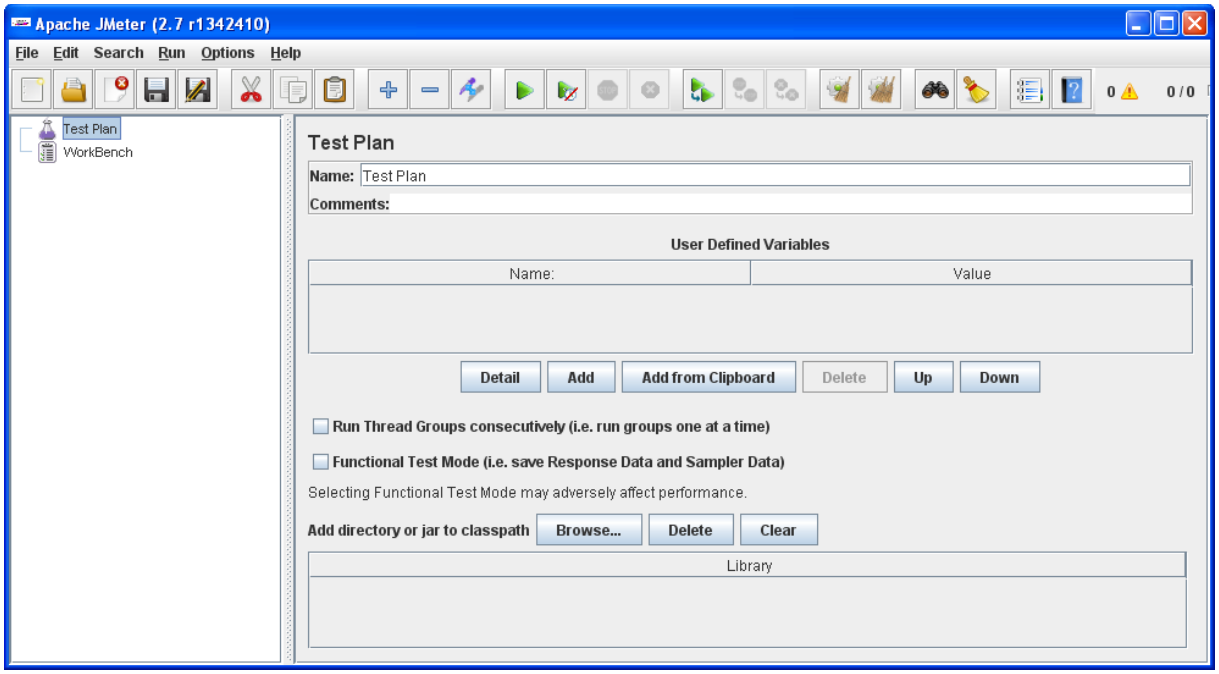


Slika 27: Pregled uporabe CPU v esxtop

Preverili smo, ali naš sistem obvladuje vse zahteve s stališča obremenitve strojne opreme. Ugotovili smo, da sta obe procesorski jedri v povprečju izkoriščeni 18% in 26%, kar kaže v splošnem na nizko obremenjenost. Ta podatek pa je lahko zavajajoč, saj so strežniki pogosto časovno neenakomerno obremenjeni. V ta namen smo pri testiranju uporabili namenska orodja, ki testirajo učinkovitost sistemov, ob tem pa spremljali izrabo sredstev celotnega sistema. Na strežniku APP1 smo uporabili orodje Apache JMeter (Slika 28), ki je izvedlo obremenitveni test s poudarkom na simulaciji izvajanja zahtev spletnih aplikacij. Orodje je sicer mogoče uporabiti na praktično vseh sistemih, ki lahko poganjajo programsko kodo v Javi. Na strežniku APP2 smo namestili preprosto, a zmogljivo orodje stress, ki se zažene kar iz ukazne vrstice:

```
stress -c 50 -m 100 -vm-stride 1024 -d 10
```

Z zgornjim ukazom smo obremenili procesor, delovni spomin in trdi disk.

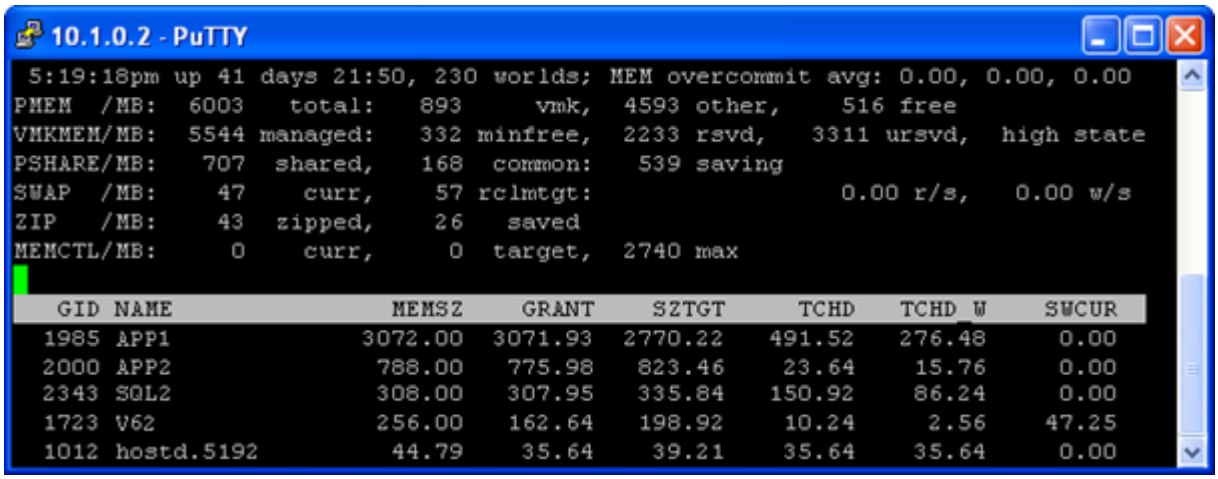


Slika 28: Aplikacija Apache JMeter

Na strežnikih SQL2 in V62 nismo pognali nobenih orodij, saj ni predvideno, da bi se na njih izvajale večje obremenitve.

Med izvajanjem smo s pomočjo orodja esxtop preverili obremenitev procesorja, ki je ostala znotraj normalnih okvirjev (42% in 68% po jedru). Porabo moči po porabniku (virtualni strežniki in procesi samega sistema ESXi) je razvidna v stolpcu **%USED**, pri čemer je seštevek vseh porabnikov enak št. procesorskih jeder × 100, **idle** pa predstavlja neporabljen delež moči. Vseeno smo preverili, ali lahko sistem nadalje optimiziramo in s tem pohitrimo izvajanje aplikacij na virtualnih strežnikih. V stolpcu **%RDY** je prikazan odstotek časa, ko kateri od porabnikov čaka, da bo procesor lahko izvršil njegovo zahtevo. Če je kateri izmed odstotkov visok, je smiselno, da temu sistemu dodelimo več jeder, drugje pa jih odvzamemo, če ugotovimo, da niso potrebni. Pri tem moramo sicer upoštevati, da pohitritve ne bomo dosegli, če sama aplikacija ni prilagojena za uporabo več jeder hkrati.

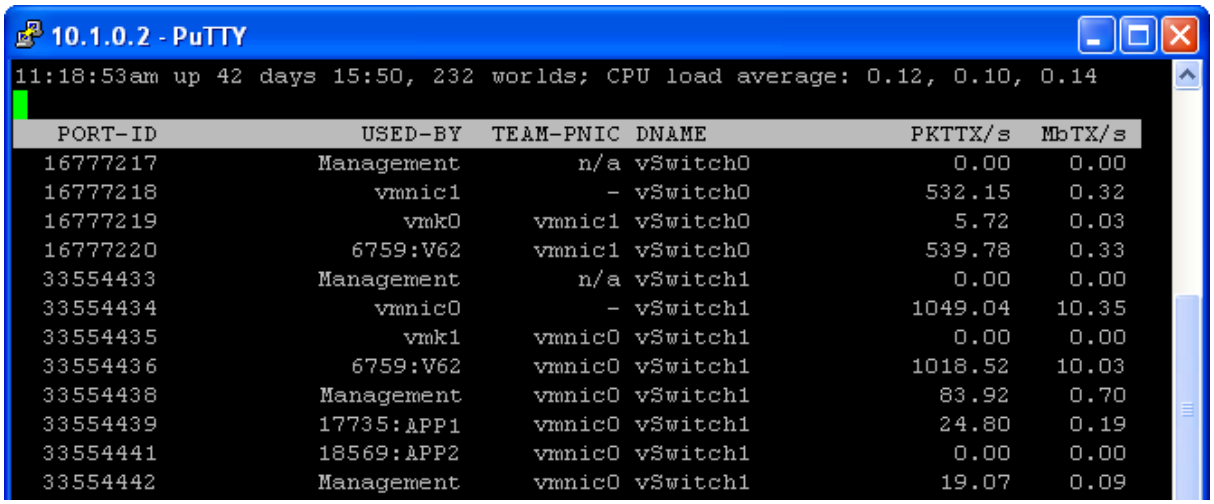
Če je izvajanje aplikacij na strežnikih časovno potratno kljub nizki porabi procesorske moči moramo razloge iskati v drugih sredstvih in komponentah (podatkovni centri, drugi strežniki ipd.) [5]. Z ukazom `m` nam esxtop prikaže porabo delovnega spomina (Slika 29).



Slika 29: Pregled uporabe MEM v esxtop

V prvi vrstici vidimo povprečje **MEM overcommit**, ki prikazuje razmerje med delovnim spominom, ki je dodeljen virtualnim strežnikom in dejanskim fizičnim spominom, s katerim upravlja ESXi, zmanjšan za 1. ESXi zna namreč z delovnim spominom upravljati tudi tako, da dodeljen, vendar neizkoriščen spomin dodeli drugemu virtualnemu strežniku, če ga ta takrat potrebuje. Našim strežnikom skupno ni dodeljenega več spomina, kot ga fizično imamo, zato je naš **overcommit** enak 0,00. V splošnem strežniki delujejo hitreje, če imajo na voljo več delovnega spomina, zato bi bilo smiselno naš sistem nekoliko optimizirati. S pomočjo tabele smo preverili, kateri izmed strežnikov bi lahko izkoristil več spomina, ter kateri ga ne uporablja v celoti. Stremeli smo k temu, da je dodeljeni spomin (**MEMSZ**) približno enak dodeljenemu (**SZTGT**), ter da je uporaba navideznega pomnilnika (**SWCUR**) enaka 0,00. Strežniku APP1 smo zato zmanjšali vrednost dodeljenega spomina, ter ga razporedili med ostale.

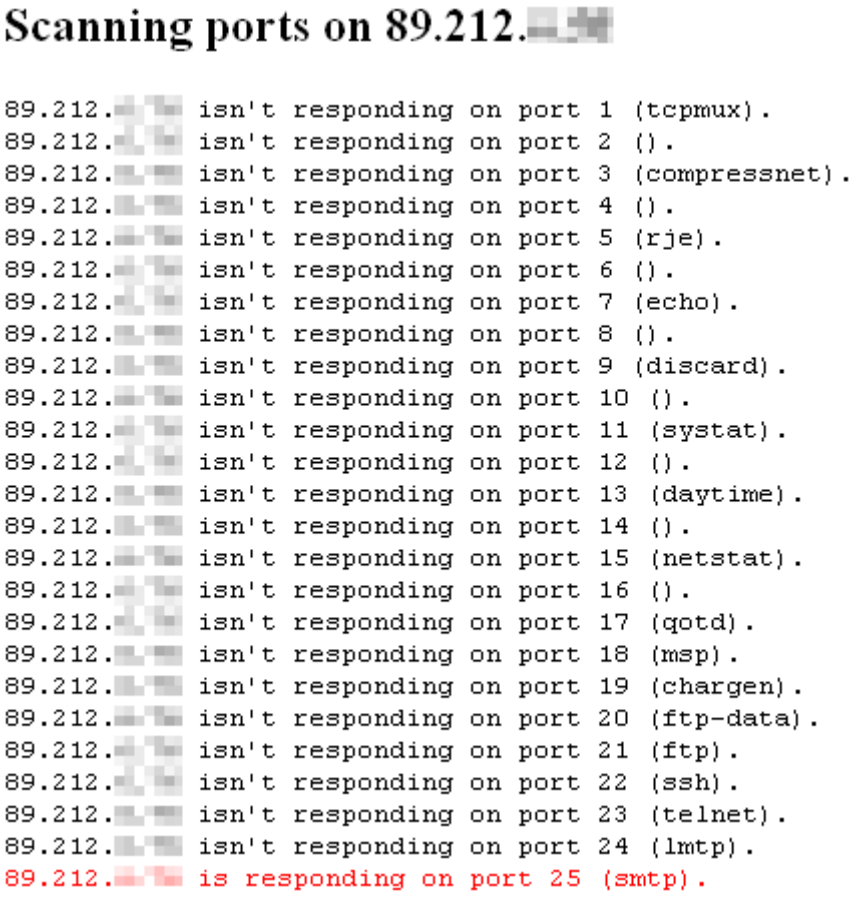
Z ukazi `d`, `u`, `v` in `n` v orodju esxtop smo preverili tudi obremenjenost podatkovnih polj, fizičnih in virtualiziranih trdih diskov ter prometa v omrežju (Slika 30). Obremenitve teh sredstev so bile nizke in ne predstavljajo ozkega grla pri izvajanju aplikacij.



Slika 30: Pregled aktivnosti omrežja v esxtop

2.6.2. Penetracijski test

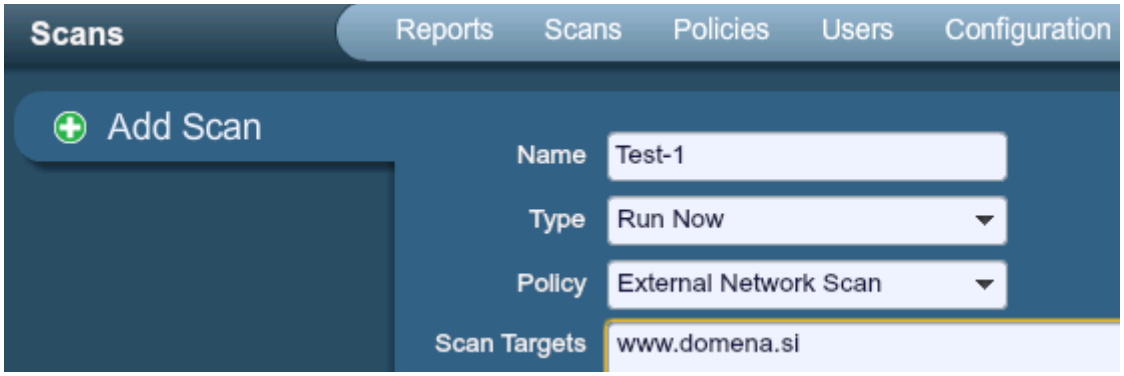
Ker je vsak sistem s povezavo v internet potencialna tarča vdora ali drugih zlonamernih dejanj smo izvedli površinski varnostni test. Sprva smo se lotili pregleda odprtih vrat in s tem preverili, ali smo požarni zid ustrezno nastavili. Spletna aplikacija[6] nam je po končanem testu prikazala seznam odprtih vrat (Slika 31).



Slika 31: Pregled odprtih vrat

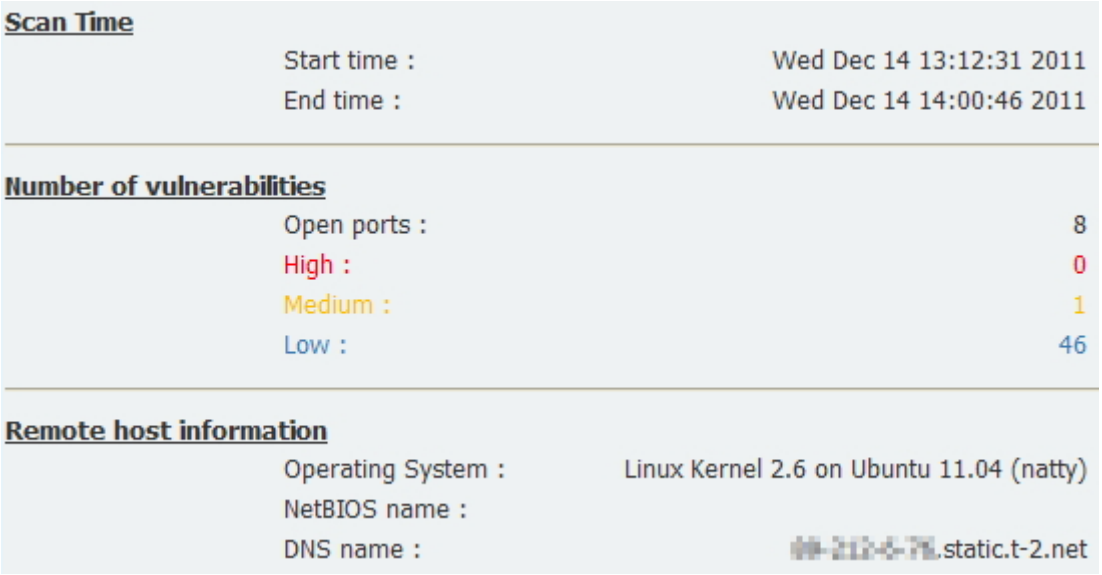
Nadalje smo se lotili bolj podrobnega testa z uporabo aplikacije Nessus[7], ki je za nekomercialno uporabo brezplačna. Le-ta pregleda sistem večnivojsko za širok spekter potencialnih ranljivosti, vključno s poskušanjem dostopa do sistemov ali storitev z uporabo privzetih in široko uporabljenih gesel. Za uporabo moramo namestitveni paket prenesti s spleta ter ga namestiti na računalnik, kar terja nekoliko več dela, kot pri nameščanju običajnih aplikacij. Potrebna je namreč registracija na spletni strani proizvajalca za pridobitev aktivacijske kode in nastavitev administratorskega uporabniškega imena in gesla. Ko je namestitev zaključena se v aplikacijo prijavimo preko spletnega brskalnika, nakar lahko že pričnemo s testiranjem. Za prvi test smo preko menija (**Scans→Add**) izbrali zunanji varnostni pregled (**External Network Scan**). V polje **Scan Targets** smo vpisali našo domeno ter test zagnali (Slika 32).

[6] http://www.t1shopper.com/tools/port-scan/
[7] http://www.nessus.org/products/nessus/



Slika 32: Varnostni pregled s programsko opremo Nessus

Ob končanem preverjanju smo podrobno poročilo o vseh zaznanih nevarnostih ter priporočili o odpravi le-teh izvozili v html obliki, ter ga podrobneje preučili. V našem sistemu je bila zaznana le ena ranljivost z oznako srednje kritičnosti ter 46 informativnih ali nizko kritičnih ranljivosti (Slika 33). Ponovno so bila preverjena tudi odprta vrata.

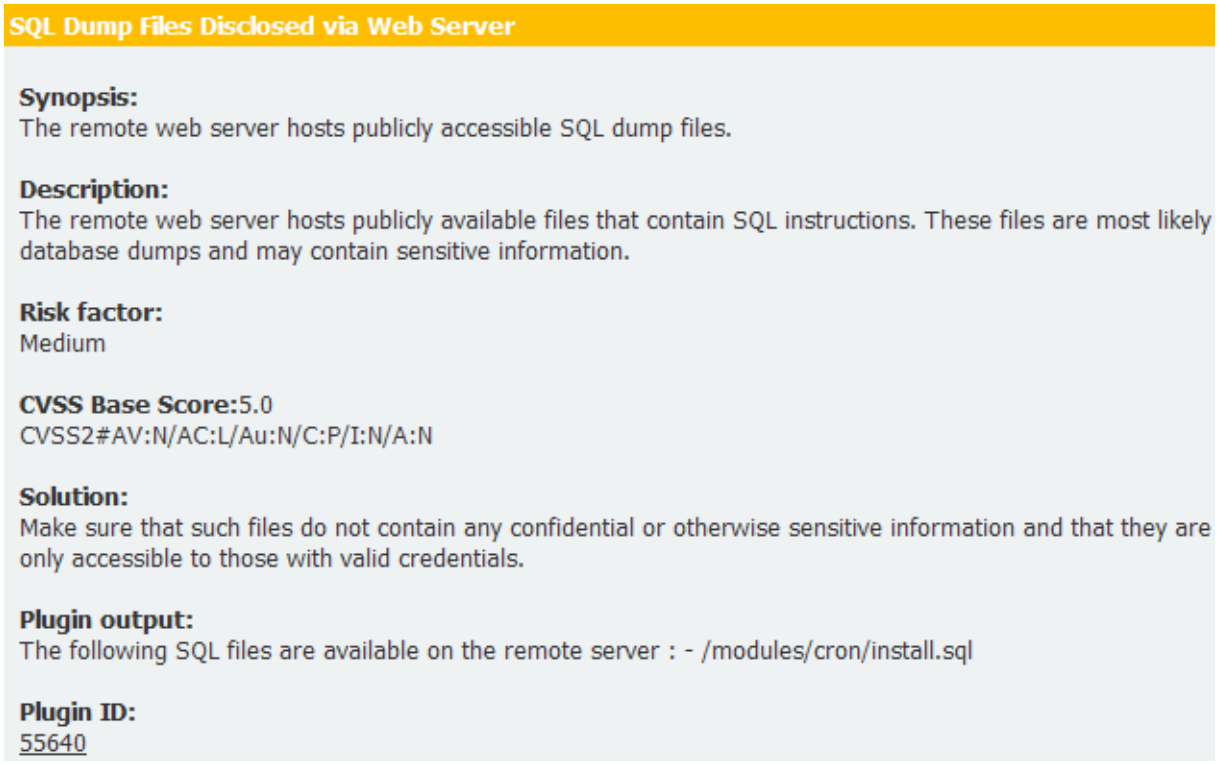


Slika 33: Pregled zaznanih ranljivosti

Poročilo je zelo podrobno in nam poleg zaznanih ranljivosti prikaže tudi vse ostale informacije, ki jih je možno pridobiti iz našega sistema, kot so verzija programske opreme ali vsebina 'robots.txt' datoteke. Prikaz teh informacij sicer ne predstavlja neposrednega tveganja, vendar pa si z njimi napadalec lahko pomaga pri napadu z uporabo socialnega inženiringa. Tovrstni napadi so vse bolj pogosti [15], zato smo se posvetili tudi tem podatkom in, če je obstajala možnost, vzpostavili njihovo nedostopnost.

Ranljivost z oznako srednje kritičnosti (Slika 34) je aplikacija Nessus zaznala zaradi datoteke .sql, ki je bila prosto dostopna. Gre za eno od namestitvenih datotek, ki vsebuje podatke o shemi podatkovne baze, ki jo neka aplikacija uporablja. V razdelku **Solution** nam je aplikacija Nessus

svetovala, kako naj to ranljivost odpravimo, kar smo storili s spremembo pravic dostopanja do mape, kjer se datoteka nahaja.



Slika 34: Pregled zaznane ranljivosti

Po odpravljenih pomanjkljivostih smo opravili ponovni pregled, saj smo le tako lahko prepričani, da je sistem res bolj varen in da s spremembami nismo povzročili kakšne nove varnostne luknje.

2.6.3. Test nastavitev DNS

Po izvedenem varnostnem pregledu smo preverili še ustreznost nastavitev DNS. Le-te so izjemnega pomena pri zagotavljanju dostopnosti naših storitev, predvsem elektronske pošte in spletnih strani. Tudi v ta namen smo uporabili spletno aplikacijo[8], kamor smo zgolj vpisali našo domeno in počakali na rezultate. Spisek vseh nastavitev je precej dolg, zato smo za namen te diplomske naloge smo prikazali le po en rezultat pravilne in nepravilne nastavitve, opozorila in obvestila (Slika 35).

[8] http://www.intodns.com/

!	Reverse MX A records (PTR)	ERROR: No reverse DNS (PTR) entries. The problem MX records are: **189.193.185.91.in-addr.arpa -> no reverse (PTR) detected** You should contact your ISP and ask him to add a PTR record for your ips
i	WWW A Record	Your www.domena.si A record is: www.domena.si [91.185.193.189]
✓	IPs are public	OK. All of your WWW IPs appear to be public IPs.
!	Different autonomous systems	WARNING: Single point of failure

Slika 35: Pregled ustreznosti nastavitev DNS

Rezultati so pokazali, da nimamo ustreznega zapisa PTR, kar smo morali popraviti, aplikacija pa je zaznala tudi šibek člen v našem sistemu. Ob izpadu strežnika DNS namreč nimamo sekundarnega strežnika, zato bi tak izpad pomenil nedosegljivost tudi ostalih storitev. Ker pa je namen našega sistem zgolj akademski in ne komercialni, je takšno tveganje sprejemljivo.

2.7. Primerjava s plačljivim sistemom

Okolje, ki smo ga preizkušali je brezplačno, na voljo pa je več možnosti licenciranja opreme ESXi. Plačljive različice nam dodatno (glede na paket) omogočajo [6] tudi:

- uporabo večjega števila procesorjev in količine spomina,
- uporabo centralnega nadzora ESX in ESXi strežnikov (vCenter),
- uporabo skript pri namestitvah,
- naprednejše možnosti izdelave varnostnih kopij in
- več drugih funkcij.

Proti plačilu se lahko odločimo tudi za plačljivo podporo sistema. Če sistem uporabljamo v podjetju, kjer se obdeluje veliko podatkov v realnem času (bančništvo, zavarovalništvo, javna uprava, policija ipd.) je to posebej priporočljivo, saj je potrebno izpad delovanja kar najhitreje odpraviti.

3. Zaključek

V diplomskem delu smo se osredotočili predvsem na uporabo brezplačnih orodij in programske opreme in dokazali smo, da je le-ta na vsaj tako visoki ravni kot plačljiva. Uspešno smo namreč vzpostavili osnoven sistem, ki lahko gosti spletne aplikacije, razrešuje naslove IP in skrbi za elektronsko pošto, hkrati pa je varnost sistema na visokem nivoju.

Vzpostavljeni sistem je tudi energetsko in stroškovno učinkovit, saj uporabljamo le en fizični strežnik, za administracijo pa je dovolj že en strokovnjak. V kolikor pa bi se odločili naše sisteme prestaviti v komercialno okolje, nam je to s selitvijo virtualnih strežnikov močno olajšano. Računalništvo v oblaku je namreč v vzponu, saj se zaradi nižjih stroškov in manjše potrebe po administraciji vse več podjetij odloča za najem storitev IT pri ustreznem podjetju, ter se raje posvetijo prodaji in odnosu do strank.

Virtualizacija se tudi nenehno vse bolj razvija, saj je podjetje VMware pred kratkim[9] na tržišču predstavilo novo različico svoje programske opreme – ESXi 5.1, ki prinaša nove in izboljšane funkcionalnosti [2].

[9] 10. septembra 2012

Viri

[1] (2012) VMware compatibility Guide. Dostopno na: http://www.vmware.com/resources/compatibility/search.php

[2] (2012) VMware vSphere Features. Dostopno na: http://www.vmware.com/products/vsphere/mid-size-and-enterprise-business/features.html

[3] (2012) Download vSphere Hypervisor for Free. Dostopno na: https://www.vmware.com/tryvmware/?p=free-esxi&lp=default

[4] (2012) VMware Guest Operating System Installation. Dostopno na: http://partnerweb.vmware.com/GOSIG/home.html

[5] (2008) Performance Analysis Methods. Dostopno na: http://www.vmware.com/files/pdf/perf_analysis_methods_tn.pdf

[6] (2012) Compare VMware vSphere Kits. Dostopno na: http://www.vmware.com/products/datacenter-virtualization/vsphere/compare-kits.html

[7] (2011) ESX / ESXi Whitebox HCL. Dostopno na: http://www.vm-help.com/esx40i/esx40_whitebox_HCL.php

[8] (2011) Vyatta Guide to Documentation. Dostopno na: http://www.vyatta.com/downloads/documentation/VC6.2/Vyatta_GuideToDocumentation_R6.2_v01.pdf

[9] (2011) Vyatta Firewall Reference Guide. Dostopno na: http://www.vyatta.com/downloads/documentation/VC6.2/Vyatta_FirewallRef_R6.2_v01.pdf

[10] (2011) Vyatta Security Reference Guide. Dostopno na: http://www.vyatta.com/downloads/documentation/VC6.2/Vyatta_SecurityRef_R6.2_v01.pdf

[11] (2012) Snort Downloads. Dostopno na: http://www.snort.org/snort-downloads

[12] (2009) The greatest open source software of all time. Dostopno na: http://www.infoworld.com/d/open-source/greatest-open-source-software-all-time-776?source=fssr

[13] (2012) Fortune 500. Dostopno na: http://www.fortune.com/500/

[14] (1987) Domain Names – Implementation and Specification. Dostopno na: http://tools.ietf.org/html/rfc1035

[15] (2004) Security Threats Toolkit. Dostopno na: http://www.social-engineer.org/wiki/archives/SEDefined/SEDefined-GreatestRisk.htm