Tartu Ülikool

Loodus- ja täppisteaduste valdkond

Tehnoloogiainstituut

Sander Kuusemets

Courses veebisaidi serveri kaasajastamine ja turvaprobleemide lahendamine

Bakalaureusetöö (12 EAP)

Arvutitehnika eriala

Juhendaja:

Kristjan Krips

Tartu 2018

Resümee/Abstract

Veebisaidi courses.cs.ut.ee serveri kaasajastamine ja turvaprobleemide lahendamine

Arvutite arenedes kasvab süsteemide ja tarkvara keerukus, seega tuleb järjest rohkem tähelepanu pöörata küberturvalisusele ja töökindlusele, eriti keskkondades, mis käsitlevad isikuandmeid või teisi andmeid, mida kasutades on võimalik isikuandmetele ligi pääseda.

Antud töö eesmärgiks on luua kaasaegne süsteem, mis oleks võimeline pakkuma turvalist serveriteenust Courses veebisaidile. Selle käigus leiti ning dokumenteeriti turvaprobleemid, pakkuti neile lahendus hea tava alusel ning kirjeldati kuidas välistada nende uuesti tekkimist.

Töö käigus valmis Tartu Ülikooli Teadusarvutuste keskuse halduses olev kõrgkäideldav koormust tasakaalustav veebiklaster.

CERCS: T120 Süsteemitehnoloogia, arvutitehnoloogia

Märksõnad: server, veebiturve, veebiserver

Updating and solving the security problems of courses.cs.ut.ee web site server

Advancement of computers allows the complexity of computer systems and software to increase. This means one should pay more attention towards cyber security and reliability. Especially in environments which either handle people's personal data or data, that can be used to obtain access to people's personal data.

The purpose of this thesis was to develop a system which is capable of providing a secure highly available load balanced hosting to the Courses web site, document security problems, provide a best-practice solution and develop a plan how to prevent these kinds of problems in the future.

As a result of this thesis, a highly available load balanced web cluster was setup in the management of the High Performance Center of University of Tartu.

CERCS: T120 Systems engineering, computer technology

Keywords: server, web security, web server

Sisukord

Konfiguratsioonifailide loetelu

Jooniste loetelu

Tähised, lühendid, definitsioonid

Suur osa definitsioonidest on Vallaste e-teatmikust eesti keelsete vastete puudumise tõttu.

Veebileht - veebileheks nimetatakse veebis (internetis) asuvat dokumenti [1].

Veebisait - Veebisait kujutab endast kodulehega (*home page*) algavat veebifailide (HTML failide) kogumikku mingil teemal. Igal veebisaidil on oma veebiaadress (*web address*), mis võib kuuluda eraisikule, firmale või organisatsioonile. Veebisaite hoitakse veebiserveritel, mida haldavad firmad ise või ISP'd. Suured veebisaidid võivad asuda ka osadena mitmel eraldi serveril [1].

PHP (*PHP: Hypertext Preprocessor*) - platvormist sõltumatu veebiserveri poolne skriptikeel, mida kasutatakse enamasti veebisaitide arendamiseks [1].

LDAP (*Lightweight Directory Access Protocol*) - Komplekt protokolle, mis võimaldavad ligipääsu infokataloogidele [1].

MySQL - Väga populaarne avatud lähtekoodiga relatsioonbaasihaldur (RDBMS) Rootsi firmalt MySQL AB, mis kasutab struktureeritud päringukeelt (SQL) [1].

Server - Tarkvara mis rakendab teenust [2].

Masin - Füüsiline või virtuaalne arvuti mis majutada ükskõik mis arvu servereid [2].

Klaster - Arvutisüsteemides nimetatakse klastriks serveritest ja muudest ressurssidest koosnevat rühma, mis funktsioneerib ühe tervikliku süsteemina ning võimaldab hõlpsat juurdepääsu ning mõnel juhul ka koormuse tasakaalustamist ja paralleeltöötlust [1].

Distributsioon - Kõigile huvilistele kasutamiseks avaldatud vabavara versioon. Sõna *distribution* e. jaotamine kasutatakse sõna *version* asemel selleks, et eristada tasuta jaotatavat vabavara raha eest müüdavast kommertstarkvarast [1].

SSH (*Secure Socket Shell, Secure SHell*) **- turvaline kest, turvakest** UNIX'i-põhine käsuliides ja protokoll, mis võimaldab turvalist sisselogimist kaugarvutisse [1].

SFTP (*SSH File Transfer Protocol*) **- turvaline failiedastusprotokoll** IETF'i platvormist sõltumatu võrguprotokoll, mis võimaldab lisaks failiedastusele üle ebaturvaliste võrkude ka eemalasuvasse arvutisse saadetud failide haldamist [1].

Ansible - Automatisatsioonisüsteem, mis käsitleb konfiguratsioonihaldust, rakenduste jaotamist, pilve provisioneerimist, ad-hoc käskude käivitamist ja masinate orkestreerimist [3].

ARP (*Address Resolution Protocol*) **- aadressiteisenduse protokoll** Protokoll IP aadressi vastendamiseks arvuti füüsilisele ehk MAC-aadressile Etherneti kohtvõrgus (Etherneti-aadressile). Näiteks IP praegu kõige levinuma versiooni IP version 4 (IPv4) puhul on IP aadressi pikkus 32 bitti, aga Ethernet'i võrgus on seadmete aadresside pikkuseks 48 bitti . Seepärast peetakse ARP-puhvri nime all tuntud tabelit, mis seab omavahel vastavusse IP-aadressid ja MAC-aadressid. ARP annab ette protokollireeglid, mille alusel toimub selle vastavuse tekitamine ja aadresside teisendamine [1].

Linux - Tasuta levitatav UNIX-i laadne operatsioonisüsteem, mis jookseb tervel real riistvaraplatvormidel, sh Intel'i ja Motorola mikroprotsessoritel. Linuxi kerneli töötas välja soomlane Linus Torvalds [1].

SFTP (*SSH File Transfer Protocol*) - **turvaline failiedastusprotokoll** IETF'i platvormist sõltumatu võrguprotokoll, mis võimaldab lisaks failiedastusele üle ebaturvaliste võrkude ka eemalasuvasse arvutisse saadetud failide haldamist. Liikumises olevate andmete turvalisuse tagamiseks kasutatakse SSH-2 protokolli [1].

IDS (*Intrusion Detection System*) - **sissetungi avastamise süsteem** Seade või tarkvararakendus, mille ülesandeks on avastada väljastpoolt lähtuvat pahatahtlikku tegevust arvutites või võrkudes. Siia kuuluvad väga erinevad süsteemid alates tavalistest viirusetõrje programmidest kuni hierarhiliste süsteemideni, mis jälgivad andmeliiklust kogu magistraalvõrgus [1]. Leidub erinevaid tüüpi süsteeme - HIDS (*Host Intrusion Detection System*), ehk masina põhine sisssetungi avastamise süsteem ja NIDS (*Network Intrusion Detection system*), ehk võrgu põhine IDS.

TLS (*Transport Layer Security*) protocol **- transpordikihi turbeprotokoll** Avatud protokoll, mis võimaldab klient-server rakendustel omavahel turvaliselt suhelda üle Interneti, olles kaitstud pealtkuulamise või sõnumite rikkumise ja võltsimise eest.
TLS-i kasutamisel toimub nii otspunktide autentimine kui ka andmeedastus krüpteeritult. Harilikult autenditakse ainult serverit ja klient jääb autentimata. Kui soovitakse autentida ka klienti, tuleb kliendile tagada PKI kasutusvõimalus. Andmeedastusel kasutatakse enamasti sümmeetrilist krüptograafiat [1]. Sümmeetrilise krüptograafia võtme kokku leppimiseks kasutatakse asümmeetrilist krüptograafiat.

Moodul - Tehnikas nimetatakse mooduliteks iseseisvaid komponente, millest masin või mehhanism on koostatud. Moodulkonstruktsiooni eeliseks on see, et ühest kindlast moodulite komplektist saab hõlpsasti koostada mitmeid erinevaid masinaid [1].

YAML (*YAML Ain't Markup Language*) - YAML on inimesesõbralik andmete jadaesituse standard kõigi programmeerimiskeelte jaoks [4].

GPFS (*General Parallel File System***)** - IBM-i poolt arendatud suure jõudlusega jagatud failisüsteem mõeldud klastrikeskkondades kasutamiseks [5].

IT (*Information Technology*) - ***infotehnoloogia*** Infotehnoloogia on termin, mis katab kõiki digitaalse informatsiooni loomise, salvestamise, edastamise, tõlgendamise ja käitlemise valdkondi [1].

NFS (*Network File System)* - **võrgu-failisüsteem** Sun Microsystems'i protokoll, mis on defineeritud standardis RFC 1094 ja mis lubab arvutile juurdepääsu failidele üle võrgu nii, nagu asuksid need failid sellesama arvuti kohalikel ketastel. See protokoll on sisse ehitatud enam kui 200 firma toodetesse ja on praegu de facto standard. NFS'i realiseerimiseks kasutatakse ühenduseta protokolli (UDP), et muuta see olekuvabaks [1].

SLA (*Service Level Agreement*) - **teenusetaseme leping** Leping teenusepakkuja ja kasutaja vahel, kus on kirjas lepingu kehtivusaja kestel oodatav teenusekvaliteet. Teenusetaseme lepinugid kasutatakse nii müüjate ja ostjate vahel kui ka sisemiselt IT-osakondade ja nende lõppkasutajate vahel. Neis lepinguis võidakse ära määrata kasutajale kättesaadav ribalaius, rutiinide ja ad hoc päringute reaktsiooniajad, probleemide lahendamise reaktsiooniajad (võrk maas, arvutirike jne) ning tehnilise personali hoiakud ja käitumismallid [1].

TCP (*Transmission Control Protocol*) - **edastusohje protokoll** Levinuim võrgu transpordikihi protokoll, mida kasutatakse Etherneti võrkudes ja Internetis.
TCP on ühendusega edastuse protokoll, mis on ehitatud internetiprotokolli (IP) peale ja seetõttu näeme lühendit TCP peaaegu alati kombinatsioonis TCP/IP ("TCP IP peal"). TCP lisab internetiprotokollile töökindla sideühenduse ja andmevoo reguleerimise ning võimaldab täisdupleksühendusi.
Teine internetiprotokolli peal käitatav protokoll on ühenduseta edastuse protokoll UDP (*User Datagram Protocol*) [1].

MAC-address (*Media Access Control Address*) - **MAC-aadress, meediumipöörduse juhtimise aadress** Kohtvõrgus (või mõnes muus võrgus) on MAC-aadress teie arvuti võrgukaardile tootja poolt omistatud unikaalne riistvaranumber. Etherneti kohtvõrgus on see identne teie ethernetiaadressiga. Kui teie arvuti on ühendatud Internetiga (IP-protokolli kohaselt on teie arvuti siis host), paneb vastavustabel teie IP aadressi vastavusse teie arvuti füüsilise MAC-aadressiga kohtvõrgus [1].

Kernel - tuum, kernel Ressursijaotust ja muid põhifunktsioone hõlmav operatsioonisüsteemi keskne osa ehk südamik. Termini "kernel" sünonüümiks on "nucleus", mis tähendab samuti tuuma. Tuuma vastandiks on kest, mis on opsüsteemi kõige välimine osa ja suhtleb kasutaja poolt antavate käskudega. Arvuti käivitamisel laaditakse kernel kõigepealt põhimällu spetsiaalselt selleks ettenähtud kohta. Kernelile reserveeritud mälupiirkond on kaitstud, nii et sinna pole võimalik midagi muud salvestada. Kuna kernel jääb põhimällu kuni arvuti väljalülitamiseni, siis peab ta olema võimalikult väike, kuid samal ajal suutma teenindada opsüsteemi kõiki ülejäänud osi ja rakendusprogramme. Tüüpiline kernel vastutab mäluhalduse (*memory management*), protsessi- ja tegumijuhtimise (*process and task management*) ning kõvaketta halduse (*disk management*) eest [1].

Port - ***võrguport*** Liides, mille kaudu saab üle võrgu pöörduda konkreetse programmi poole. Võrgupordid on harilikult nummerdatud ning võrguprotokollid (näit. TCP või UDP) lisavad edastatavatele andmepakettidele koos sihtkoha IP-aadressiga ka pordinumbri. IP-aadress määrab ära, millisele arvutile andmeid saadetakse ja pordinumber näitab, millisele programmile need lähevad. IP-aadressi ja pordinumbri kombinatsiooni nimetatakse sokliks [1].

Round Robin - **järjestikplaanur** Koormusjaotuse algoritm, mille alusel valitakse iga ressurss järjest ühekaupa, kuni kõik ressursid on läbi käidud ja alustatakse algusest [1].

ISP (*Internet Service Provider*) **- internetiteenuste pakkuja, internetipakkuja** ISP on firma, mis pakub eraisikutele ja teistele firmadele ligipääsu Internetile ja sellega seotud teenustele nagu näit. veebisaitide ehitamine ja hostimine.
ISP omab Interneti võrgupunktile vajalikke seadmeid ja ühendust sidevõrkudega selles geograafilises piirkonnas, mida ta teenindab. Suurematel ISP'del on oma kiired püsiühendused, nii et nad sõltuvad vähem sideteenuste pakkujatest ja suudavad tagada oma klientidele paremat teenust [1].

Regex (*Regular Expression*) **- regulaaravaldis** Regulaaravaldis on string, mis kirjeldab või langeb kokku mingi stringide hulgaga vastavalt kindlatele süntaksireeglitele. Paljud tekstiredaktorid ja utiliidid kasutavad regulaaravaldisi otsingute tegemiseks ja teksti käitlemiseks vastavalt etteantud mustritele. Regulaaravaldisi toetavad paljud programmikeeled. Näit. Perl'i ja Tcl puhul on võimsad regulaarvaldiste mootorid juba süntaksisse sisse ehitatud [1].

MITM (*Man In The Middle*) attack – **vahendajarünne** Rünnak, kus kahe otspunkti vahelist ühendust kuulatakse pealt või mõjutatakse pahatahtliku kolmanda poole poolt andmete kogumiseks või võõra informatsiooni sisestamiseks.

Sissejuhatus

Internet on tänapäeval kujunenud vahetu informatsiooni allikaks, mida kasutatakse igapäevaselt inimeste poolt otsuste ja valikute tegemiseks. Selline usaldus määrab teatavad kohustused veebilehtede ja -saitide omanikele - ligipääsetavuse, tervikluse ja kaasaegsuse kohustused. Need eeldavad ajaga kaasas käimist, ning aktiivset probleemide otsimist ja lahendamist omanike poolt.

Tartu Ülikool kasutab internetti tudengite, töötajate ja koostööpartneritega suhtlemiseks, õpetamiseks ja informatsiooni jagamiseks. Üks selline kasutus on populaarne veebisait Courses kus Arvutiteaduse Instituudi poolt hoitakse kursuste materjale nii tudengite kui ka avalikkuse jaoks.

Käesoleva lõputöö eesmärk oli luua kaasaegne ja turvaline veebiklaster Courses veebilehe jaoks, tagades sellega kiirema, ligipääsetavama ja turvalisema veebiteenuse. Selle käigus dokumenteeriti Courses veebisaidi vajadused, nõuded ja kitsaskohad.

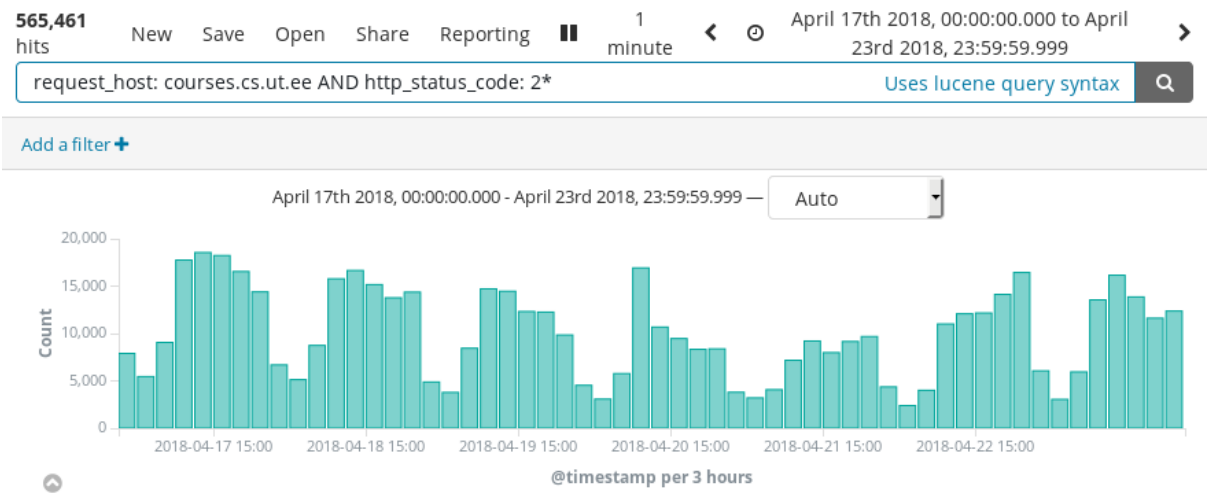
Lõputöö koosneb kuuest peatükist. Esimeses peatükis antakse ülevaade Courses veebisaidist ja selle eripäradest. Teises peatükis räägitakse vanast veebiserveri masinast ja selle põhilistest probleemidest. Kolmandas peatükis kirjeldatakse ära uue veebiklastri nõuded. Neljandas peatükis seletatakse lahti uue klastri ehitus, topoloogia ja kasutatud tehnoloogiad. Lisaks kirjeldatakse kuidas ja mis konfiguratsiooniga moodulid üles seati. Viiendas peatükis dokumenteeritakse suuremad turvaprobleemid ja pakutakse neile lahendus. Kuuendas peatükis kirjeldatakse veebiklastri ja veebisaidi seiremeetodid.

1. Ülevaade

Courses veebisait on PmWiki sisuhaldussüsteemi baasil arendatud veebisait Arvutiteaduste Instituudi tarbeks. Seal hoitakse instituudis pakutavate kursuste informatsiooni semestrite kaupa, instituudis toimuvate ürituste informatsiooni, avaliku osalemisega kursuste materjale ja vanade kursuste arhiive. Arhiivid ulatuvad tagasi 2004 aastasse.
Järgnev peatükk kirjeldab Courses veebilehte – selle vajadust, tööd ja ülesehitust.

1.1 Courses veebisaidi tutvustus

Arvutiteaduste Instituudi jaoks on Courses kriitilise tähtsusega, sest suure osa vajalikest materjalidest saavad tudengid kätte just kasutades seda veebisaiti. Õppejõududel on kohustus arhiveerimise mõttes lisada iga semester enda kursuse materjalid Courses veebisaidile, et need oleksid tulevikus kättesaadavad ka järgnevatele põlvedele.
Kuna Courses veebisait on tähtsal kohal Arvutiteaduste Instituudis, on selle kasutus ka võrdlemisi suur. Nädalas teenindab Courses veebisait umbkaudu 500 tuhat õnnestunud päringut (kahega algav HTTP staatuse kood) [6].



Joonis 1 Courses veebisaidi päringute arv nädalas

Alternatiivina pakub Tartu Ülikooli Infotehnoloogia osakond tsentraalselt teenust Moodle. Moodle on sarnase eesmärgiga veebisait mõeldud kursuste läbi viimiseks ja nende materjalide hoidmiseks, mida pakutakse kogu Tartu Ülikoolile. Moodle aga kahjuks Arvutiteaduste Instituudi (ATI) vajadusi ei rahulda. Kuigi neid teenuseid kasutatakse tihti paralleelselt, on Arvutiteaduste Instituudis õppimas ka tudengeid kellel pole Tartu Ülikooli kasutajanime ja parooli, seega on vajalik süsteem kuhu ATI saab ise lisada kasutajaid, või pakkuda kogu kursuse informatsiooni ilma autentimisvajaduseta. Samuti ei toeta Moodle kursuste ajalugu, kus tihti ühe kursuse jaoks läbi aastate on ainult üks kursus, mida siis iga kursuse lõppedes tühjendatakse ja uuendatakse. Courses veebilehel on aga saadaval “Kursuste arhiiv” [7], mille kaudu saab kätte kursuste ajaloo alates aastast 2004.

1.2 Veebisaidi ülesehitus

Courses kasutab järgmiseid mooduleid:

- PHP skriptimiskeelt serveripoolse funktsionaalsuse lisamiseks
- MySQL andmebaasi andmete hoidmiseks
- Tartu Ülikooli tsentraalseid LDAP servereid autentimiseks
- Apache veebiserverit veebisaidi sisu serveerimiseks

Courses veebisait hõivab 200 gigabaiti salvestuspinda, koosneb 521715 failist ja sellel on peaaegu 130 tuhat erinevat URL-i.

Number of dynamic URLs:	25468
Number of static URLs:	103006
Total number of parameters:	63009
Number of unique parameter names:	78

Joonis 2 Burp Suite veebi kraapimise tulemus

Courses asub Arvutiteaduste Instituudi virtuaalmasinas domeeninimega demeter.at.mt.ut.ee.

2. Vana veebiserveri masin Demeter

Arvutiteaduste Instituudi virtualiseerimiseks mõeldud füüsiliste masinate peal jookseb virtuaalmasin domeeninimega demeter.at.mt.ut.ee. See virtuaalmasin pakkus Courses veebilehele veebiserveri teenust aastast 2004 kuni 19. detsembrini 2017. Kuigi Demeter masinal on korduvalt operatsioonisüsteeme vahetatud, on nüüdseks see siiski ajale jalgu jäänud.

Alates 2017 aasta lõpust on Courses veebileht kolitud uue veebiserveri teenust pakkuva süsteemi peale. Uued virtuaalmasinad asuvad Demeter-iga samas klastris, aga kuna uus süsteem vajab vähemalt nelja virtuaalmasinat, siis on uue süsteemi virtuaalmasinad kõrgkäideldavuse saavutamiseks jaotatud mitme füüsilise masina peale laiali.

2.1 Tehniline kirjeldus

- Domeeninimi: demeter.at.mt.ut.ee
- IP-aadress: 193.40.36.81
- Tüüp: KVM Virtuaalmasin
- CPU: 4 Virtuaalset QEMU tuuma
- RAM: 12 GB
- Operatsioonisüsteem: GNU/Linux CentOS 5.11
- Kernel: 2.6.18-416.el5
- Veebiserver: Apache 2.2.3
- MySQL server: MySQL Server 5.0.95
- MTA: Postfix 2.3.3
- Tulemüüri kiht: iptables
- Veebilehtede failisüsteem: NFS-iga Arvutiteaduste Instituudi GPFS-i pealt külge võetud

2.2 Puudujäägid

Demeter masina põhiliseks puudujäägiks on selle masina vanus. Ajaloo jooksul on seda virtuaalmasinat liigutatud erinevate füüsiliste masinate vahel, et parandada jõudlust, turvalisust ja kättesaadavust. Kahjuks on virtuaalmasina tarkvaraline sisemus püsinud juba aastaid uuendamata (umbes 2014. aastast), kuna see tähendaks serveri tarkvara uuendamist tarkvaralisele tasemele kus vanad veebisaidid, mis Demeter masina peal jooksevad, ei sobiks kokku uute tarkvara versioonidega ja seega need veebisaidid lakkaks töötamast.
Veebisaitide arendamine ja uuendamine ei ole süsteemiadministraatorite tööülesannete hulgas, ning tihti ei ole neil ka oskust, huvi ega aega, et seda teha. Samas aga kasutaja ei soovi kulutada aega ja ressurssi, et uuendada teenust, mis praegusel hetkel töötab. Seega on jäänud Demeter masin olukorda, kus serveri- ja teenuse tarkvara ei saa uuendada, aga samas IT maailm liigub edasi, ehk võimalike turvaprobleemide hulk ainult kasvab.
Vanusest tulenevalt on masinas veel veebiteenuseid, mis on pikka aega hooldamata. Need on kunagi ammu üles seatud, siiamaani vajalikud, kuid turvaprobleemidega. See aga mõjutab ka teisi veebilehti ja -teenuseid selles masinas, kuna veebilehed ja -saidid pole veebiserveri kontekstis eraldatud. Veebilehed näevad üksteise faile, mis tähendab, et rünnete seisukohast kõik serveris olevad veebiteenused jagavad üksteise turvavigu, sest läbi ühe veebisaidi turvaprobleemi saab ligi ka teistele veebisaitidele.

2.2.1 Kriitilised probleemid

Kriitilisteks probleemideks nimetame tänapäeva headest tavadest erinevaid sätteid, teenuste versioone ja konfiguratsiooni, mis põhjustavad või lubavad ründevektoreid, mille abil saaks Courses teenuse või Demeter masina töö peatada, või Courses teenusest andmeid lugeda ilma vastavat ligipääsuõigust omamata. Veebiserveri masinat uurides ja testides leiti järgmised probleemid:

- Sisselogimise õigused olid 74-l unikaalsel TÜ tsentraalsest autentimissüsteemist tuleval kasutajal. Kui kellelgi nendest 74-st inimesest peaks parool lekkima (emaili, pahavara, MITM-i kaudu), siis pääsetakse kõigile Courses failidele ligi vähemalt lugemisõigustes.

- Courses veebisaidi failid olid suures osas ebasobivates õigustes - lubati lugemise ligipääsu ka teistele (inglise keeles *other*) peale omaniku ja grupi, k.a. autentimise logidele.

- Konfiguratsioonifailist sai lugeda andmebaasi ühenduse andmeid. Failis oli kirjas kasutaja, kellel olid lisaks kõigele muule ka UPDATE, DELETE ja DROP õigused andmebaasis. Nende õigustega saab väga lihtsalt tekitada andmekadu.
- Autentimise logidest sai lugeda autenditud kasutajate kasutajanimesid ja emaili aadresse.
- Mitmetele õppejõududele oli lubatud Courses veebisaidi failidele ligipääs nii, et nad võisid veebisaidi kaustadesse lisada enda faile. Nendeks failideks võis olla ka PHP kood, mida siis läbi veebibrauseri sai käivitada. Selline käitumine lisas veebisaidile programmeerijate ja süsteemiadministraatorite poolt tundmata koodi ja funktsionaalsust.
- Kuna õppejõududel ei ole ülevaadet lehe funktsionaalsusest ja tihti ka headest tavadest, olid nad võimelised tekitama turvavigu.
- Demeter masinasse oli paigaldatud ainult PHP 5.3.3, mille turvaprobleemide paikamine lõpetati 14. augustil 2014.
- PmWiki rakenduse logid olid läbi veebiserveri loetavad kogu maailmale - seal võis vealogides peituda salajast informatsiooni.
- CentOS 5.11 lõpetas uuenduste saamise 31. märtsil 2017 [8].

```
root@demeter:~# cat /etc/security/access.conf | grep "ALL" | grep "^+" | sed 's/+://g' |
sed 's/:ALL//g' | grep -v "LOCAL" | tr ' ' '\n' | sort | uniq -c | wc -l
74
```

Joonis 3 Ekraanitõmmis unikaalsete kasutajate kokkulugemisest

2.2.2 Väiksema kriitilisusega probleemid

- PHP versiooni uuendamine oli tehnilistel põhjustel võimatu - seda tehes oleksid kõik teised Demeter serveris asuvad lehed töö lõpetanud.
- Uuenduste ja muudatuste tegemiseks pidi ajutiselt Courses veebisaidi töö lõpetama.
- PHP protsesse käivitati veebiserveri õigustes.
- Veebiserveril puudus veebirakenduste tulemüür (inglise keeles *web application firewall)* [9].

3. Uue veebiklastri nõuded

Nõuded on erinevatest allikatest tulevate soovide, vajaduste ja ettepanekute kohandus üheks terviklikuks reeglite nimekirjaks, millest teenuse püsti panekul lähtuda otsuste tegemiseks.

Veebiserveri teenuse kontekstis mõeldakse kasutaja all inimest, kes soovib seada üles, muuta, uuendada või arendada veebilehte või -saiti. Inimest, kes kasutab veebilehte või -saiti läbi veebibrauseri või HTTP päringute nimetame kliendiks.

Järgnevates alapeatükkides on kirjeldatud nõuded, millele uus veebiklaster peab vastama.

3.1 Instituudi nõuded

Uue veebiklastri eesmärgiks on pakkuda Arvutiteaduste Instituudile süsteemi, kus saab Courses veebisaiti, kuid ka muid tähtsaid veebisaite, turvaliselt ja stabiilselt hoida. Veebiklaster peab olema loodud nii, et see oleks võimalikult veakindel ning peab suutma jagada koormust, et oleks võimalik seda tulevikus vajadusel laiendada. Lisaks sellele peab veebiklaster olema jälgimise all piisavalt hea seiramise tehnoloogiaga, et oleks võimalik aru saada, kui miski on valesti.

3.2 Funktsionaalsed nõuded

Funktsionaalsed nõuded kirjeldavad süsteemi tööd ning seda, mida süsteem peab tegema.

- Peab saama lisada, muuta ja eemaldada kasutajaid.
- Kasutajatele peab saama pakkuda talle vajalikku PHP versiooni hetkel toetatud versioonide [10] vahel.
- Veebisaidid peavad saama vajadusel saata e-maile.
- Veebisaitide ja -lehtede töö peab olema seiratud, et tuvastada halba käitumist.
- Kasutajad peavad olema loogiliselt eraldatud, et vältida üksteise häirimist.
- Veebiteenus peab toetama liikluse krüpteerimist TLS näol.

3.3 Mittefunktsionaalsed nõuded

Mittefunktsionaalsed nõuded on süsteemsed kvaliteedimõõtmed, mis defineerivad teenusekvaliteedi (inglise keeles *Quality of Service*), ning seda kuidas süsteem peab toimima. Mittefunktsionaalsed nõuded arendavad arhitektuuri [11].

Kasutaja tase

- Kasutatavus - Kasutaja peab saama minimaalse Linuxi oskusega (SFTP/SSH protokollide abil) üles laadida ja sättida PHP toel töötava veebilehe või -saidi kasutajale antud Linuxi kasutaja õiguste piires.

Teenuse tase

- Töövõime - Süsteem peab olema võimeline serveerima ära kõik kliendid ka tipptundide ajal ilma ühenduste kaotamiseta puuduliku jõudluse või mahutatavuse (inglise keeles *capacity*) tõttu.
- Usaldusväärsus - Süsteemil ei tohi olla üksikut veakohta (inglise keeles *single point of failure)* Teadusarvutuste keskuse poolt kontrollitavate ressursside piires (moodulid, masinad, failisüsteemid).
- Kättesaadavus - Ametlik kättesaadavus (SLA) Teadusarvutuste keskuse kohustuste järgi on 95%, ehk ~18 päeva lubatud rikkeaega aastas. Sisemiselt sihime 99.9%, ehk ~9 tundi rikkeaega aastas [2].

Strateegiline tase

- Skaleeritavus - Süsteem peab olema lihtsasti skaleeritav teenuste töö katkemiseta. Peab olema võimeline lisada mooduleid, masinaid ja teenuseid ilma, et kättesaadavus kannataks.
- Paindlikkus - Erinevad moodulid peavad olema piisavalt eraldatud, et nende asendamine ei nõuaks mõne teise mooduli väljavahetamist.

Süsteemi tase

- Turvalisus - Turvalisus peab olema piisavalt kõrgel tasemel iga asjakohase komponendi jaoks, et autoriseerimata ligipääs on võimatu v.a. juhtudel kus selleks panustatakse ebaproportsionaalselt palju resurssi, mis puhul peavad olema rünnakute vektorid tuvastatavad.
- Laiendatavus - Kehtivad samad reeglid, mis skaleeritavuse jaoks.
- Hallatavus - Süsteemi käitumine peab olema kaetud seiretehnoloogiatega piisavalt hästi, et vajalikud faktorid otsustusprotsesside jaoks oleksid kaetud. Nendeks on läbilaskvus, tehingute arv sekundis ja tõrgete arv ajas.
- Hooldatavus - Kõik hooldusvajadused peavad olema automatiseeritud konfiguratsioonihalduse või seiretehnoloogiate abil.
- Stabiilsus - Süsteemil ei tohi olla üksikut veakohta (inglise keeles *single point of failure)* Teadusarvutuste keskuse poolt kontrollitavate ressursside piires (moodulid, masinad, failisüsteemid).

3.4 Teadusarvutuste Keskuse headest tavadest tulenevad nõuded

Teadusarvutuste Keskusel on kogemusest ja positsioonist tulenevalt hulk nõudeid, mis peavad olema täidetud, et tagada süsteemide töökindlus ja haldamise lihtsus. Need nõuded ei ole absoluutselt kohustuslikud, kuid nendest kinnipidamine teeb teiste süsteemiadministraatorite töö lihtsamaks ning suurendab Teadusarvutuste Keskuse tegevuste läbipaistvust instituudi piires.

Teadusarvutuste Keskuse jaoks on tähtis, et kogu tehtud töö oleks dokumenteeritud tasemeni kus teine süsteemiadministraator võiks dokumentatsiooni alusel püsti panna samasuguse süsteemi. Selleks dokumentatsiooniks võib kasutada nii Infotehnoloogia Osakonna teenust wiki.ut.ee, või Ansible-t.

Dokumentatsioon on tähtis, sest Teadusarvutuste Keskuse varukoopiate tegemise poliitika alusel ei tehta varukoopiaid masinatest ja süsteemifailidest, vaid ainult tähtsatest andmetest. Süsteemi taastamine näeb välja nii, et vajadusel seatakse dokumentatsiooni või automaatikat kasutades vajalikud masinad ja serverid töökorda ja siis kopeeritakse teenuste andmed tagasi õigesse kohta. Seejärel peaks teenus olema jälle töövõimeline.

Tähtis on ka kasutada võimalikult palju vabavaralisi ja avatud lähtekoodiga teenuseid. Vabavaralise teenuse puhul jääb rohkem raha keskusele, mida saab investeerida paremasse riistvarasse või töötajatesse. Avatud lähtekood pakub võimalust iseseisvalt otsida lähtekoodist vigu, ning suurendab potentsiaalsete kommertseesmärkidel infot koguvate osade või turvavigade keerukust, kuna avalikkusel on võimalus koodi ja selle kvaliteeti kontrollida, mistõttu on lihtsamate probleemide ülesleidmine tõenäolisem.

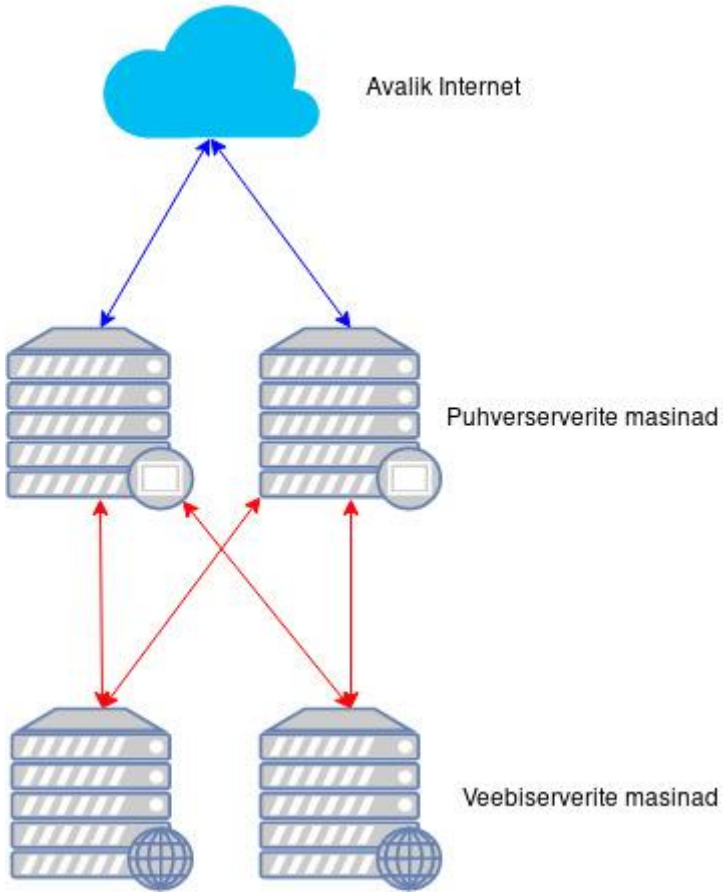
4 Uue veebiklastri kirjeldus ja teostus

Eelnev peatükk kirjeldas ära erinevad nõuded, mis on seatud uuele veebiklastrile ning selle teenustele. Järgnevas peatükis kirjeldatakse ära uue veebiklastri topoloogia ja teenused, mis igas masinas jooksevad ning kuidas neid üles seada.

4.1 Süsteemi topoloogia kirjeldus

Lähtudes headest tavadest ja seatud nõuetest, on kõrgkäideldavuse jaoks vajalik, et iga masin ja komponent oleks vähemalt dubleeritud. See tähendab, et näiteks veebiserver peab olema vähemalt kahes masinas, aga mõlemad serverid peavad olema võimelised ära teenindama tavalise ühenduste hulga, muidu on tegu ainult koormusejaotusega ja kõrgkäideldavuse vajadused pole täidetud.
Tulenevalt koormusjaotuse ja turvalisuse nõuetest tuleks süsteemi topoloogia etteotsa lisada ka proksiserver, läbi mille kõik saadetud HTTP päringud edastatakse õige veebiserverini. See lubab veebiserveri masinad ära peita ülikooli sisevõrku nii, et internetist pole neile ligipääsu. Lisaks võimaldab see seadistada logimist tsentraalses kohas ning rakendada kõigile päringutele standardsed turbereeglid veebiserverite seadistusest, versioonist või implementatsioonist sõltumata. Kõrgkäideldavuse eesmärgil dubleerime ka proksiserveri.

Järgnev pilt kirjeldab terve klastri arhitektuuri virtuaalmasinate tasemel.



Joonis 4 Masinate topoloogia (sinine joon – välisvõrk, punane joon – sisevõrk)

4.2 Konfiguratsioonihaldus

Uus veebiklaster koosneb mitmest erinevast masinast ja teenusest, mille konfiguratsiooni ja sisu peab hoidma omavahel kooskõlas. Lühikeses perspektiivis on seda võimeline tegema ka inimene (süsteemiadministraator), aga inimesed teevad vigu. Vead hakkavad kumuleeruma, ning see tekitab probleeme pikas perspektiivis.

Üks lahendus selleks on väga täpselt ära dokumenteerida tehtud tööd ning töövood. See aga keeruliste süsteemide puhul põhjustab probleemi, kus keegi ei taha seda teha, sest keerukus põhjustab veaohtu ja liigne dokumenteerimine on süsteemiadministraatorile ebavajalikult kurnav.

Sellise probleemi lahenduseks on loodud automatisatsiooni tehnoloogiad, mis on võimelised ette antud reeglistiku järgi masinates ülesandeid täitma. Ülesandeid nagu pakettide

installimine, teenuste üles seadmine, konfiguratsioonifailide kopeerimine ja muutmine jms. Selliseid tööriistu on mitmeid, kuid Teadusarvutuste keskuse heade tavade alusel kasutatakse veebiklastri üles seadmiseks Ansible-t.

Ansible lubab defineerida YAML faili kujul reegliraamatu (inglise keeles *playbook*), kus on järjestikku kirjas Ansible moodulid [12] koos nende moodulite argumentidega, masinate põhise konfiguratsiooniga ja muutujatega, mida käivitamisel kasutatakse. Nende argumentide alusel käivitab Ansible masinates ülesandeid, mis kontrollivad masina hetkeseisu ja kui see seis erineb sellest, mis mooduli argumentides kirjas, siis käivitab käskusid, et viia selle masina seis selliseks.

```
- name: Yum | Install necessary packages
  yum:
    name: "{{ item }}"
    state: present
  with_items:
  - httpd24u
```

Loetelu 1 Näide Ansible "Yum" mooduli definitsioonist – paigaldab veebiserveri paki pakihaldurist

Veebiklastri üles seadmiseks loodi neli erinevat reegliraamatut erinevateks eesmärkideks.

1. „web-frontend.yml“, seab üles proksiserverite masinad
2. „web-backend.yml“, seab üles veebiserverite masinad
3. „add_site.yml“, lisab uue kasutaja veebiklastrisse
4. „config-sync.yml“, sünkroniseerib proksiserverite ja veebiserverite konfiguratsiooni versioonihalduse omaga

Kolmanda reegliraamatu puhul on ka oluline märkida, et selle käivitamiseks on vaja defineerida muutujaid Ansible sees – *username*, *php_version*, *domain*, *enable_mail* ja *SSL*. Nende muutujate väärtused tulenevad kasutaja nõuetest kasutajanimele, domeeninimele, emaili vajadusele, PHP versioonile ja TLS-ile.

Kõik vajalikud ülesanded saab süsteemiadministraator ära teha nende Ansible reegliraamatutega – uuendamised, kasutajate lisamised ja eemaldamised, konfiguratsiooni muutmised. Selline struktuur lubab ka väga lihtsasti laiendada klastrit – tuleb ainult lisada

masin veebiklastri definitsiooni Ansible konfiguratsioonis õige rolliga (proksiserver või veebiserver), Ansible käivitada ja kui see lõpetab, siis on masin liitunud klastriga.
Ansible reegliraamat on inimestele suhteliselt hästi loetav, kuna see on YAML formaadis. See lubab Ansible-t kasutada ka ühe dokumentatsiooni vormina, kus on kirjas kuidas midagi masinas tehti ja sätestati. Siis peab igaks ülesandeks aga kasutama ainult Ansible-t ning masina sees käsitsi ei tohi konfiguratsiooni muuta. Selline käitumine põhjustaks olukorra kus mingid liigutused ei ole dokumenteeritud ja see võib hiljem tekitada probleeme Ansible jooksutamisega.

4.3 Proksiserverite masinates kasutatud tehnoloogiad

Eelmistes alampeatükkides kirjeldati, kuidas veebiklaster näeb välja masinate tasemel – kaks proksiserverit, kaks veebiserverit, erinevad ühendused nende vahel. Selgitati ka konfiguratsioonihalduse vajadusest ning sellest kuidas see teostatud on. Järgnevalt kirjeldatakse proksi masina sisemusest, milliseid tehnoloogiaid ja teenuseid kasutatakse ning kuidas neid üles seada.

4.3.1 HAProxy

HAProxy on vabavaraline proksiserver, mis on võimeline vahendama TCP ja HTTP protokolli kasutavaid päringuid. See tarkvara on mõeldud olema võimalikult lihtsalt implementeeritav ja töökindel, samas aga toetab ka keerulisi lahendusi kui on vaja teha midagi väga spetsiifilist [13].

HAProxy eelisteks teiste populaarsete proksiserverite ees on konfiguratsiooni võimsus ja lihtsus ning proksiserveri kiirus ja turvalisus . Squid serverit kasutatakse rohkem kui on vaja lisaks proksimisele ka puhverdamist. Nginx ei ole lõplikult vabavaraline, mitmed Nginx funktsionaalsused vajavad Nginx Plus paketti, mis on tasuline [14].

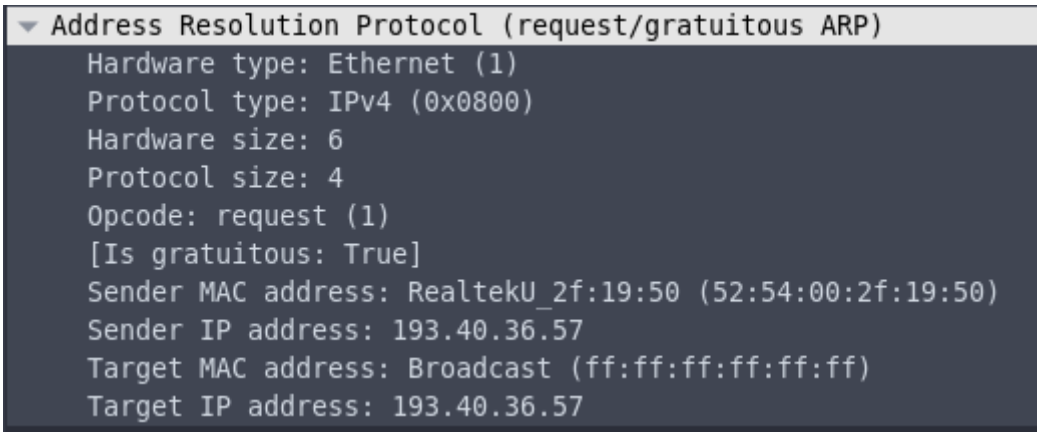
4.3.2 PCS

PCS (inglise keeles *Pacemaker/Corosync Configuration System*) on kahest erinevast süsteemist kokku pandud serveriklastrite loomise ja haldamise teenus. Need kaks komponenti on Pacemaker ja CoroSync. Pacemaker on teenuste ja võrgu orkestreerimiseks kasutatav tarkvara, mis suudab vastavalt reeglitele käivitada ja seisata teenuseid ja virtuaalseid võrguaadresse. Corosync kontrollib klastri erinevate liikmete kättesaadavust ja sünkroonis

olekut. Lisaks hoiab Corosync töös kvoorumit, millega hoitakse ära lõhenenud aju (inglise keeles *split brain*) olukorda. Koos töötades suudavad need teenused liigutada teenuseid ka klastri liikmete vahel, lubades paremat kõrgkäideldavust ja isegi geograafilist liiasust.

4.3.3 Gratuitous ARP

Pärast virtuaalse IP-aadressi liigutamist ühe masina küljest teise külge ei ole sellest liigutusest teadlik vastava võrgu eest vastutav võrguvärav, mistõttu läbi võrguvärava liikuvaid pakette suunatakse ikkagi vana võrguvärava ARP tabelis puhverdatud MAC-aadressini. Seetõttu kasutab PCS gARP (inglise keeles *gratuitous ARP*) päringu vastust, kus saatev masin seab paketi allika ja sihtkoha IP-aadressid enda omadeks, allika MAC-aadressiks uue seadme MAC-aadressi ja sihtkoha MAC-aadressiks laialipaisu (inglise keeles *broadcast*) aadressi. See sunnib teisi võrgus olevaid masinaid uuendama enda ARP tabeli sisu, seostades omavahel virtuaalse IP-aadressi ja uue masina võrguliidese MAC-aadressi. Pärast võrguvärava ARP tabeli uuendamist teab võrguvärav, mis MAC-aadressile peab paketid saatma ja virtuaalse IP-aadressi töö jätkub. Juhul, kui kogu protsess toimub kiiremini kui viimase TCP paketi taasedastamine, ei saa serveriga ühenduses olev klient aru, et IP-aadressi liigutamine on toimunud, sest paketikadu ei teki.



Joonis 5 Gratuitous ARP päring

4.3.4 Firewalld

Firewalld (inglise keeles *Firewall Daemon*) on põhiliselt CentOS, Red Hat Enterprise Linux ja Fedora Linuxi distributsioonide poolt kasutatav tulemüüri abstraktsioonikiht tavalise linuxi kerneli põhise tulemüüri "iptables" otsas. Firewalld teeb lihtsamate tulemüürikonfiguratsioonide haldamise ja automatiseerimise mugavaks, kuna sellega saab reegleid lisada teenuse tasemel ning pakub paremat liidestust mitme võrguliidese puhul. Lisaks on firewalld halduskäsud idempotentsed.

4.3.5 Poliisipõhine marsruutimine

Poliisipõhine marsruutimine on tehnika mida kasutatakse pakettide suunamiseks vastavalt eeldefineeritud marsruutimise reeglitele. Enamasti kasutatakse selleks, et sisse tulevatele pakettidele ja ühendustele oleks võimalik vastata mööda sama marsruuti.

4.4 Seadistamine

Eelnev peatükk kirjeldas tehnoloogiaid mida kasutatakse proksiserverite üles seadmisel. Järgnevalt kirjeldatakse kuidas proksiservereid üles seada koos konfiguratsioonihalduse ja konfiguratsioonifailide näidetega.

Mõlemal proksiserveril on kaks võrguliidest, üks liides Tartu Ülikooli avalikus võrgus, mis on ligipääsetav kogu maailmast, ja teine liides sisevõrgus. Selline eraldamine on vajalik, et teised samas avalikus võrgus olevad masinad ei saaks pealt kuulata proksiserverite ja veebiserverite vahelist liiklust. Võrguaadresse haldab ülikooli DHCP server.

Ressursi raiskamise vältimiseks ja koormusjaotuse võimekuse kasutamise jaoks pannakse proksiserverid ülem-ülem süsteemi, kus tavaolukorras mõlemad proksiserveri masinad vastavad päringutele.

4.4.1 Järjestikplaanur DNS

Selleks, et klientide päringud jõuaksid mõlema serverini, peab klientide arvutitele selgeks tegema, et server vastab kahe IP-aadressi peal. Selleks saab nimeserverites sättida ühte domeeninime lahenema kahe IP-aadressi peale.

```
web.cs.ut.ee.     IN   A     193.40.36.55
                  IN   A     193.40.36.57
```

Loetelu 2 veebiklastri domeeninime konfiguratsioon nimeserveris

Selline definitsioon tähendab, et igal nimeserveri päringul valitakse nimekirjast IP-aadress järjestikplaanur algoritmi alusel, mis tagab koormuse jaotumise IP-aadresside vahel.

4.4.2 Poliitikapõhine marsruutimine

Linuxi põhised operatsioonisüsteemid vaikimisi saadavad võrguühenduste vastused mööda vaikelüüsi. Proksiserverite puhul tuleb MITM rünnakute vältimiseks kliendiga suhtluse paketid hoida rangelt avaliku virtuaalse IP-aadressi liidese peal ja veebiserverite liiklus sisevõrgu liidesel. Seega tuleb üles seada poliitikapõhine marsruutimine.

Selleks tuleb teha kaks kerneli marsruutimise tabelit, üks iga võrguliidese jaoks. Näiteks “avalik” ja “sisemine”. Seejärel tuleb sisemise võrguliidese jaoks defineerida järgmised reeglid:

```
iif ens3 table sisemine
from <sisemine IP-aadress> table sisemine
```

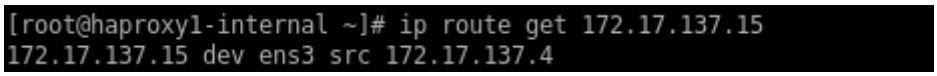
Loetelu 3 /etc/sysconfig/network-scripts/rule-ens3 võrgureeglite fail

```
<sisevõrk> dev ens3 table sisemine
default via <gateway> dev ens3 table sisemine
```

Loetelu 4 /etc/sysconfig/network-scripts/route-ens3 marsruutimise fail

Sama tuleb teha ka avaliku võrgu võrguliidese jaoks, kuid kuna avaliku võrgu liideste peal liiguvad IP-aadressid ringi kõrgkäideldavuse eesmärgil, peavad ka reeglid kaasa liikuma. Seetõttu tuleb seda teha kasutades PCS tehnoloogiat.

Marsruutimise töö kontrollimiseks saab kasutada Linux-i käsku “ip route get <aadress>”, mis tagastab IP-aadressi ja võrguliidese, mida kasutatakse vastava aadressi poole pöördumiseks.



Joonis 6 Marsruuti kontrollimise tulemus

4.4.3 HAProxy

HAProxy tarkvara saab paigaldada enamuste populaarsemate Linuxi distributsioonide pakihalduse abil. Selleks, et seejärel HAProxy tööle panna on vaja kopeerida HAProxy konfiguratsioonifail asukohta */etc/haproxy/haproxy.cfg*.
HAProxy tähtsad osad on kaks *frontend*-i mis kuulavad ühendusi maailmast portidel 80 ja 443, ja suunavad ühendused õigesse *backend*-i. Pordil 443 kuulav ühendus vastab TLS kasutavatele ühendustele.

```
frontend main
     bind *:80
     default_backend default

frontend main-ssl
     bind *:443 ssl crt /etc/ssl/ ca-file /etc/ssl/CA/CA.crt
↳ no-sslv3
     default_backend ssl
```

Loetelu 5 HAProxy "frontend"-ide konfiguratsioon

Backend-id on samamoodi jagatud, üks *backend* ühendub veebiserveriga kasutades HTTP protokolli, teine kasutades HTTPS-i.

```
backend default

      option forwardfor
      option httplog

      balance static-rr
      cookie SERVERID insert indirect nocache httponly

      server web1 172.17.137.10:80 check cookie web1
      server web2 172.17.137.11:80 check cookie web2


backend ssl

      option forwardfor
      option httplog

      balance static-rr
      cookie SERVERID insert indirect nocache httponly secure

      server web-ssl1 172.17.137.10:443 check ssl verify none cookie
web1
      server web-ssl2 172.17.137.11:443 check ssl verify none cookie
web2
```

Loetelu 6 HAProxy "backend"-ide konfiguratsioon

TLS verifikatsioon on välja lülitatud, kuna see ei ole proksiserveri ja veebiserveri vahelisel suhtlusel vajalik, kuna veebiserver kasutab ise genereeritud sertifikaati ilma sertifitseerimisasutuse sertifikaadita. Ise genereeritud sertifikaate on vajadusel lihtsam ja kiirem vahetada.

4.4.4 PCS

Tulenevalt DNS-i nõrkustest ei tohi koormusejaotuse ja kõrgkäideldavuse puhul lootma jääda ainult *Round Robin* DNS tehnoloogiale. DNS kirjeid puhverdatakse nii ISP kui ka operatsioonisüsteemi tasemel. Mitmetel firmadel on ka enda DNS serverid kus võib toimuda omakorda puhverdamine. Kui üks nimeserverites defineeritud IP-aadressidest lõpetab töö, siis suure tõenäosusega kasutab kliendi arvuti katkist IP-aadressi edasi, mistõttu tuleb kasutada PCS tehnoloogiat, et virtuaalne IP-aadress töötava serveri peale üle tõsta.

```
pcs cluster auth <proksimasin 1> <proksimasin 2>
pcs cluster setup --name veebiklaster <proksimasin 1> <proksimasin
2>
pcs cluster start --all
```

Misjärel peaks töötama PCS klaster. Seda saab kontrollida järgmise käsuga:

```
pcs status
```

Kui eelnev käsk veateadet ei tagasta, siis on klaster töös. Seejärel tuleb teha mõned olukorra-spetsiifilised muudatused. Kvoorumit saab moodustada alates kolmest masinast, seega see tuleb välja lülitada. Lisaks, ilma kvoorumita ei saa teha tarastamist (inglise keeles *fencing*), sest veaolukorras üritavad mõlemad masinad üksteist isoleerida, mis peataks teenuse töö.

```
pcs property set stonith-enabled=false
pcs property set no-quorum-policy=ignore
pcs resource defaults resource-stickiness=100
```

Virtuaalsete IP-aadresside ja teenuste lisamiseks tuleb kasutada ressursiagente (inglise keeles *resource agents*) [15].

```
pcs resource create ClusterIP-57 ocf:heartbeat:IPaddr2
ip=193.40.36.57 cidr_netmask=23 nic=ens9 op monitor interval=1s

pcs resource create ClusterIP-55 ocf:heartbeat:IPaddr2
ip=193.40.36.55 cidr_netmask=23 nic=ens9 op monitor interval=1s

pcs resource create PBR ocf:heartbeat:Route destination=0.0.0.0/0
table=avalik device=ens9 gateway=193.40.36.1 op monitor
timeout="20" interval="1"

pcs resource create HAProxy systemd:haproxy op monitor
interval=1s
```

Seejärel tuleb kloonida teenused, mis peaksid töötama mõlema masina peal. Kloonimine aitab hoida tavaolukorras teenust mõlema masina peal töös, kuid ühe masina kadumisel ei pandamõlemat klooni sama masina peale.

```
pcs resource clone PBR
pcs resource clone HAProxy
```

Järgnevalt tuleb defineerida prioriteedid ja eelistused erinevate ressursside jaoks, et IP-aadressid püsiksid erinevate masinate küljes.

```
pcs constraint location ClusterIP-55 prefers <proksimasin 1>
pcs constraint location ClusterIP-57 prefers <proksimasin 2>
pcs constraint order ClusterIP-55 then HAProxy-clone
pcs constraint order ClusterIP-57 then HAProxy-clone
pcs constraint order ClusterIP-55 then PBR-clone
pcs constraint order ClusterIP-57 then PBR-clone
```

Samuti sunnib see teenuste tööd alustama järgnevas järjekorras, kõigepealt IP-aadress, siis muud teenused. Seda seetõttu, et ilma IP-aadressita ei saa muuta marsruuti (tuleb Linux-i veateade) ja HAProxy ei hakka hiljem lisandunud IP-aadressi peal teenindama.

4.4.5 Firewalld

Veebiserverid kasutavad standardi järgi porti 80, et teenindada HTTP protokolli kasutavaid kliente ja porti 443, et teenindada HTTPS protokolli kasutavaid kliente. Proksiserverid teesklevad klientide jaoks veebiserverit, seega tuleks tulemüüris lubada uued ühendused portidele 80 ja 443.

```
- name: Add firewalld rules
  firewalld: port={{item}}/tcp permanent=true state=enabled
  with_items:
  - "{{ ports }}"

- name: Restart firewalld
  service:
      name: firewalld
      state: restarted
      enabled: yes
```

Muutuja *ports* defineeritakse inventarifaili sees vastavalt masina vajadustele. Siinsel juhul niimoodi:

```
ports="[80, 443]"
```

4.5 Veebiserverite masinate komponendid

4.5.1 Apache

Apache HTTP server on vabavaraline lahtise lähtekoodiga veebiserveri tarkvara. Hetkel on see kõige populaarsem veebiserveri tarkvara [16]. Seda tarkvara on pikalt kasutatud, mistõttu on see robustne ja töötab väga paljude operatsioonisüsteemide peal. Apache serverit on enamuste Linuxi distributsioonide peal võimalik paigaldada paketihaldusest.

4.5.2 PHP-FPM

PHP FastCGI protsessihaldur on veebiserverist eraldatud PHP skriptide käivitamise mootor. PHP-FPM-i kasutatakse põhiliselt suure koormusega veebiserverite puhul, kuna see lubab eraldada veebiserveri ja PHP mootori loogika ja ressursid.

4.5.3 Memcached

Memcached on vabavaraline lahtise lähtekoodiga objektide puhverdamise süsteem. Seda kasutatakse veebisaitide kiirendamiseks, et vähendada aeglasemate infoallikate kasutamise vajaduse tihedust. Memcached hoiab räsitabelit puhverdatud objektidest mälus, sellega kõiki sisend/väljund operatsioone (inglise keeles *input/output operations*) kiirendades.

4.5.4 NFS

NFS on jagatud võrgu-failisüsteemi protokoll, mis lubab arvutile juurdepääsu failidele üle võrgu nii, nagu asuksid need failid sellesama arvuti kohalikel ketastel.

4.5.5 Auditd

Auditd on Linuxi Auditeerimise Süsteemi kasutajaruumi (inglise keeles *userspace*) poolne komponent. See vastutab auditeerimislogide kättesaadavuse eest - kirjutab neid kettale, lubab neid otsida ja nende abil raporteerida.

4.5.6 SELinux

SELinux (inglise keeles *security enhanced linux*) on kerneli modifikatsioonidest ja kasutaja ruumi tööriistadest koosnev Linuxi kerneli turbemoodul, mis implementeerib MAC (inglise keeles *Mandatory Access Control*) süsteemi. SELinux defineerib ligipääsu ja kasutusõigused iga kasutaja, rakenduse, protsessi ja faili jaoks süsteemis, mida kasutatakse nende üksuste interaktsioonide kontrollimiseks kasutades konfigureeritud turbepoliitikaid, kus on kirjas kui range kontroll peab olema.

4.6 Seadistamine

Eelnev peatükk kirjeldas tehnoloogiaid mida kasutatakse veebiserverite üles seadmisel. Järgnevalt kirjeldatakse kuidas veebiserveri masinaid üles seada koos konfiguratsioonihalduse ja konfiguratsiooni failide näidetega.
Mõlemal veebiserveri masinal on võrguliides eraldi alamvõrgus, mis on ülejäänud ülikooli sisevõrgust eraldatud.

4.6.1 NFS

Masinate kõrgkäideldavuse ja koormusjaotuse vajaduse tõttu peab ka failisüsteem olema kõrgkäitlemiseks võimeline. Kuna mõlemad veebiserverid on HAProxy konfiguratsioonis kirjas ülemus-ülemus suhtes, nagu näha loetelus 6, siis peab olema veebiserveritel, rakendustel ja kasutajatel võimalik mõlema masina sees failisüsteemile samaaegselt kirjutada.
NFS-i lubamiseks Linuxi serverile on kõigepealt vaja teenindavas NFS serveris lubada klientserveri ligipääsu failisüsteemile. Selleks tuleb lisada */etc/exports* faili järgnevad read:

```
/jagatav/kaust <Veebimasina IP>(rw,async,no_root_squash)
```

Loetelu 7 /etc/exports sisu NFS võimekuse jaoks

Pärast seda tuleb käivitada käsk, mis sunnib jagavat serverit */etc/exports* faili uuesti lugema ja muudatusi sisse viima NFS-i serveri tööse:

```
exportfs -a
```

Seejärel saab veebimasinate poole peal failisüsteemi külge võtta, lisades */etc/fstab* faili vastavad read:

```
<NFS serveri domeenimimi>:<jagatav kaust> /projects nfs
defaults,nfsvers=3 0 0
```

Loetelu 8 /etc/fstab sisu NFS võimekuse jaoks

Kataloog */projects* on siinpuhul veebimasina failipuus olev asukoht kuhu NFS failisüsteem külge võetakse. Protsessi lõpetamiseks tuleb käivitada käsk:

```
mount -a
```

Pärast eelmist käsku on failisüsteem kättesaadav */projects* kataloogi alt veebiserveri masina sees. Seda tuleb korrata iga veebiserveri masina kohta. Sellesse kataloogi tehakse veebimasina kasutajate kodukataloogid, nii et need kodukaustad oleks kõigi seda NFS failisüsteemi külge võtvate masinate vahel sünkroonis.

4.6.2 PHP-FPM

PHP-FPM on lihtne meetod kuidas pakkuda sama serveri piires mitut erinevat PHP versiooni. Iga veebisaidi jaoks käivitatakse erinev PHP-FPM protsesside kogumik (inglise keeles *process pool*) millel on üksteisest erinev konfiguratsioon, jõudlus (samaaegsete lõimede arv, PHP serverite arv) ja vajadusel ka PHP versioon. Lisaks pakub PHP-FPM isoleeritust erinevate kogumike vahel.

PHP-FPM iga kogumiku jaoks on vaja konfiguratsioonifaili. Näitena Ansible-s kasutatud mall PHP-FPM konfiguratsiooni jaoks, mis tekitatakse uue kasutaja lisamisel veebiklastrisse asub lisas 1. Need failid asetatakse */opt/remi/phpXX/root/etc/php-fpm.d/* kausta, kus XX tähistab soovitud PHP versiooni kausta.
Kasutaja PHP versiooni muutmise jaoks tuleb kogumiku konfiguratsioonifail viia õige versiooniga kogumiku konfiguratsioonifailide hulka, siis pärast PHP-FPM serveri taaskäivitust hakkab veebisait tööle teise PHP versiooniga.

4.6.3 Apache

Veebiserveri rolliks on võimaldada ligipääsu staatilistele failidele nagu HTML, JavaScript, CSS kui ka pildid ja videod. Lisaks suunab veebiserver päringud PHP lehtedele edasi PHP-FPM mootorisse.
Veebiserver töötab ainult ühe kasutajana. Õiguste eraldamist veebiserveri tasemel ei ole vaja, sest veebiserveri töö on ainult lugemisoperatsioonid veebilehtede ja -saitide kataloogipuust. See tähendab, et veebiserveri kasutajale tuleb anda ainult lugemisõigused kasutajate veebikaustadesse. Kirjutamisoperatsioonide täitmiseks kasutatakse PHP-d.
Apache (omab nime „httpd“ CentOS distributsioonis) veebiserveri üles seadmiseks on vaja kahte konfiguratsioonifaili - Apache üleüldist ja veebisaidi virtuaalhosti (inglise keeles *virtualhost*) konfiguratsiooni. Igal virtuaalhostile on võimalik panna erinev konfiguratsioon, aga haldamise ja automaatika huvides tuleks seda hoida võimalikult sarnasena. Ansible loob virtualhostide konfiguratsioonifaile malli alusel ja need asetatakse faili */etc/httpd/conf.d/vh_XX.conf*, kus XX asendatakse kasutajanimega. Mallide näited asuvad lisades 2 ja 3.

4.6.4 Memcached

Kuna veebisaitide failid asuvad NFS failisüsteemil, mis sõltub võrgukiirusest ja -latentsist, võib see failisüsteem kohati aeglaseks jääda, eriti kui sellele failisüsteemile teha väikseid kirjutamise ja lugemise operatsioone (muutujad, sessioonid, küpsised). Selle vastu aitab puhverdamine, kuid kõige tüüpilisem puhverdamislahendus on kirjutada puhverfailid kettale, mis siin olukorras lahendab probleemi ainult ajutiselt. Niikaua kuniks veebiserveri masinatel jagub mälu mida operatsioonisüsteem saab NFS-i puhverdamiseks kasutada ja kuniks teised puhverdamisele kuuluvad ressursid teisi puhvrist välja ei lükka.

Memcached-i abil on aga võimalik paluda puhverdatud ressurssi hoida reserveeritud mälus, mis on ideaalne asukoht tihti loetavate ressursside jaoks. Memcached ise ei toeta kõrgkäideldavust või koormusjaotust, aga seda toetavad enamused teekid ja programmid, mis Memcached-i külge ühendavad kasutades kahte ühendust korraga. See küll ei tähenda, et mõlemad Memcached serverid alati sünkroonis on - kui üks server mõndade kirjutamisoperatsioonide ajal on kättesaamatu, siis kirjutatakse teise serverisse ja kui esimene tagasi tuleb, siis ei ole mehhanismi, mis need serverid automaatselt sünkrooni viib. Küll aga pole see puhverfailide puhul oluline, neid kirjutatakse pidevalt juurde ja üle, tänu millele on tagatud kunagine sünkroonsus [17] (inglise keeles *eventual consistency*).
Memcached puhverserveri käivitamiseks CentOS distributsioonis vajalik järgnev fail:

```
PORT="11211"
USER="memcached"
MAXCONN="1024"
CACHESIZE="1024"
OPTIONS="-l <serveri IP>  >> /var/log/memcached.log 2>&1"
```

Loetelu 9 /etc/sysconfig/memcached konfiguratsioonifail

Memcached-iga on võimalik liidestuda PHP rakenduse tasandil. Sellisel juhul peab liidestuse tagama veebisaidi looja, kuna see eeldab PHP koodi poolset integratsiooni.

4.6.5 Firewalld

Veebiserverid kasutavad standardi järgi porti 80, et teenindada HTTP protokolli kasutavaid kliente ja porti 443, et teenindada HTTPS protokolli kasutavaid kliente [18]. Seega tuleks tulemüüris lubada uued ühendused portidele 80 ja 443.
Lisaks tuleb lubada masinate vaheline suhtlus pordil 11211 Memcached jaoks. Memcached-ile peavad ligi pääsema mõlemad serverid, aga mitte miski muu.

```
  - name: Add Firewalld rules for Memcached
    firewalld:
      rich_rule: >
        rule family="ipv4" source
address="{{hostvars[item]['ansible_default_ipv4']['address']}}/24
" port port="11211" protocol="tcp" accept
      immediate: yes
      permanent: yes
      state: enabled
      with_items: "{{ groups['web-backend'] }}"
```

Loetelu 10 Ansible skripti väljavõte Memcached-i lubamiseks läbi Firewalld

4.7.6 SELinux

SELinux tehnoloogial on kaks osa, millega tuleb arvestada enne, kui veebiklastri kõigil komponentidel lubatakse SELinux-i poolt enda eesmärki täita. Esimeseks osaks on eeskirjade muutmine, mille alusel SELinux teeb otsuseid kas kindal protsessil mingit käitumist lubada (näiteks veebiserveril algatada ühendusi välismaailma). Teiseks osaks on failide, protsesside ja kasutajate piiramine kindlatesse kontekstidesse ja domeenidesse, et kontrollida protsesse, nende käitumist ja nende failide kasutust ja ligipääsu.

Eeskirju saab muuta defineerides eeskirja reeglid, kompileerides need mooduliks ja lisades selle mooduli SELinuxisse. Kuna aga paljud eeskirjad kattuvad erinevate kasutajate ja serverite vahel, on SELinux-isse tehtud ka nimekiri eeskirju kontrollivatest kahendväärtusega muutjatest (inglise keeles *boolean*), mille moodulid tulevad SELinux-i endaga kaasa [19]. See teeb enamkasutatud moodulite lisamise lihtsamaks. Nende muutujate väärtusi muudetakse käsuga:

```
~]# setsebool -P <muutuja nimi> on|off
```

Loetelu 11 SELinux muutujate muutmise näide

Klastri töö jaoks on vajalik sisse lülitada järgmise nimega muutujad:

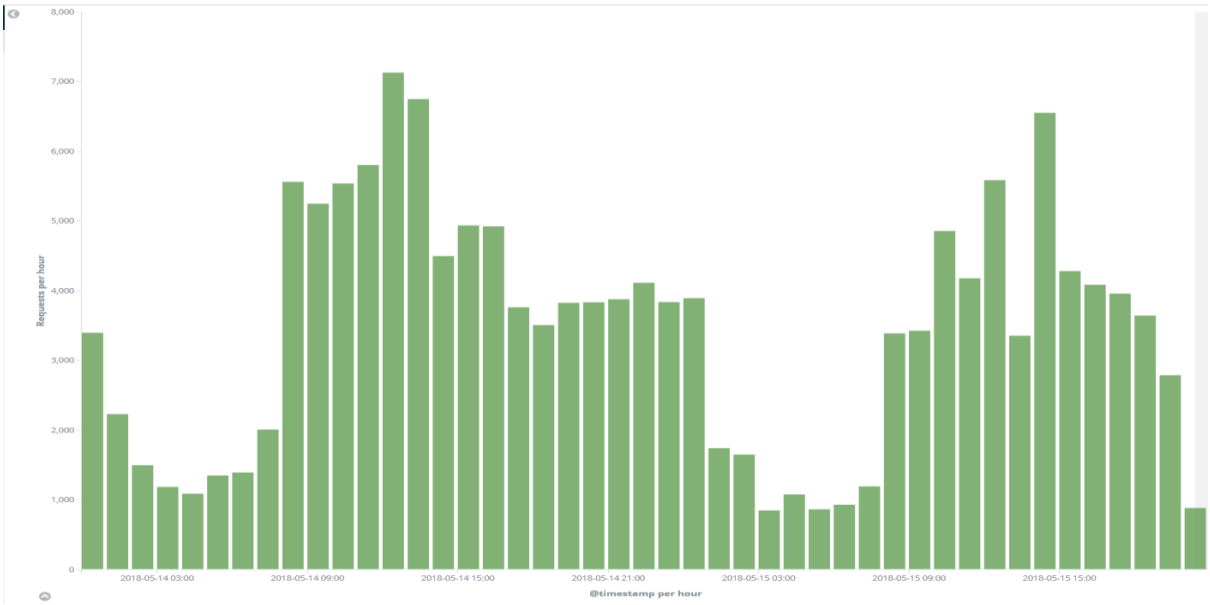
- “httpd_can_network_connect”, lubab alustada veebiserveril võrguühendusi
- “httpd_can_network_connect_db”, lubab veebiserveril ühenduda andmebaasidega
- “httpd_can_sendmail”, lubab veebiserveril saata e-maili
- “httpd_read_user_content”, lubab veebiserveril kasutada faile kodukaustadest
- “httpd_can_network_memcache”, lubab ühendada memcached serveritega
- “use_nfs_home_dirs”, lubab hoida kodukaustu NFS-i peal

Protsesside, failide ja kasutajate kontekstidesse piiramiseks kasutatakse märgistamist kus igale failile defineeritakse kolm silti - kasutaja domeeni nimi, rolli nimi ja tüübi nimi [20]. SELinuxit vaikimisi toetavatel Linuxi distributsioonidel tuleb pakihaldusest koos tarkvaraga kaasa ka enamasti õige sildistus, ehk klastri tööle saamiseks ei pea silte muutma.

4.7 Testimine

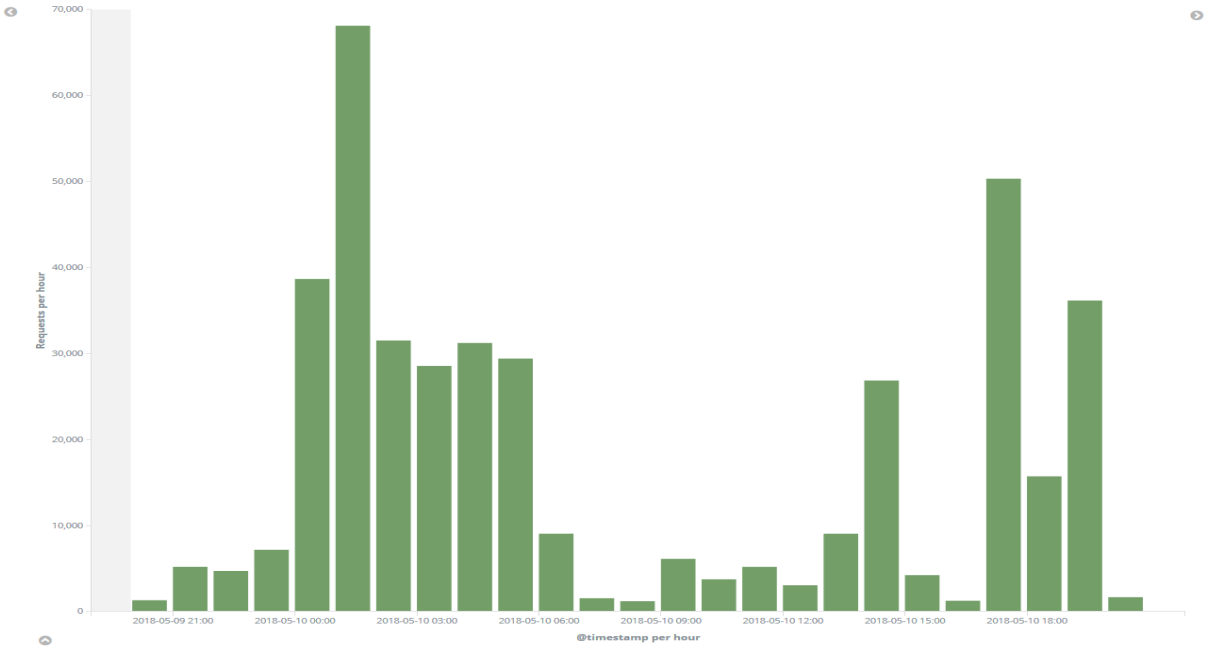
Testimise eesmärgiks on veenduda kas teenused töötavad nagu vaja, ning kas mittefunktsionaalsed nõuded on täidetud. Testimisjuhtumid peavad kajastama reaalseid olukordi. Testida tuleks jõudlust ning masinate ja serverite tõrkekindlust.

Courses veebisaidi pihta tehakse päevastel tipptundidel pea 8000 HTTP päringut.



Joonis 7 Päevane Courses veebilehe päringute arv tunni kohta

Serverid peavad olema võimelised teenindama vähemalt kaks ja pool korda nii palju päringuid, et ära teenindada kõik kasutajaid ebatavaliselt suure koormusega aegadel (semestrite esimesed päevad, eksamite eelsed päevad) ning olukordades kus mõni serveritest või masinatest ei tööta. Koormustest tehti *Apachebench* tööriistaga [21], kasutades kahtekümmet erinevat füüsilist masinat ja suvalisi URL-e Courses veebisaidilt, üritades jäljendada reaalset olukorda. Tulemus näitab, et jõudlusvajadused on rahuldatud.



Joonis 8 Koormustesti õnnestunud päringute arv tunni kohta

Kõrgkäideldavuse testimiseks segati serverite ja masinate normaalse toimingu ajal nende tööd erineva aggressiivsusega:

- lõpetati masinate tegevus järsult
- lõpetati serverite tegevus järsult
- veebimasinatel lõpetati üksikute moodulite töö
- katkestati masina töö kerneli paanikaga
- segati masinate vahelist võrku

Proksiserver suudab teise masina IP-aadressi üle võtta sekundi kuni kolmega, olenevalt lahtiste ühenduste arvust ja kui palju aega viimasest elusoleku kontrollist möödas on. See tähendab, et kõige halvemas olukorras võivad mõned TCP paketid aeguda. Selline olukord juhtub kümnest korrast ühel, ülejäänud kordadel klient muutusest aru ei saa. Kuna absoluutne kõrgkäideldavus ei ole eesmärgiks, siis see täidab eesmärgi [2].

Töötav proksiserveri masin võttis katkise masina IP-aadressi endale igas olukorras kus masina töö kompromiteeriti, välja arvatud olukorras kus proksiserverite masinad ei saanud üksteisega võrgu teel suhelda, et kontrollida teise elus olekut. Sellises olukorras on võimatu teada, mis seisus teine masin on, mis tähendab, et proksiserverite masinad üritasid võtta teise masina IP-aadressi enda külge, tekitades võrgus segaduse. Selline olukord saab aga tekkida ainult

olukorras kus terve avalik võrk on teadmata olukorras, mis blokeeriks masinatele võrgust ligipääsu niikuinii.
Veebiserverite korrasolekut kontrollivad HAProxy serverid pidevalt, ning keelduvad marsruutimast päringuid veebiserveriteni mille tervisekontroll ei õnnestu. Selleks võib kuluda kuni kaks sekundit, enne kui HAProxy server veebiserveri vigaseks kuulutab. See aga toimib ainult olukorras kus veebiserver vastab veakoodiga. Testimise käigus avastati, et kui PHP server on välja lülitatud ja PHP faile ei serveerita korralikult, aga veebiserver töötab nominaalselt, siis HAProxy ei kuuluta seda veebiserverit vigaseks ega lõpeta selle serveri pihta päringute marsruutimist. Selle probleemi parandamiseks muudeti ära veebiserveri teenuse fail Linuxi teenuste haldajas *systemd*. Selleks seoti PHP serverid ja veebiserver kokku nii, et kui ükski neist teenustest lülitub välja, lülituvad välja ka teised teenused.

```
[Unit]

BindsTo=php54-php-fpm.service   php56-php-fpm.service   php71-php-
fpm.service php72-php-fpm.service
```

Loetelu 12 Konfiguratsioonifail /etc/systemd/system/httpd.service.d/override.conf

Testimise tulemuste põhjal saab järeldada, et veebiklastril on lubatud kaotada üks veebiserver ja üks proksiserver samaaegselt, ning siiski jätkata klientide teenindamist ilma eriliste ebameeldivuseta Courses veebisaidi klientide jaoks.

5. Turvalisus

Veebisaitidel on turvaprobleemide oht väga suur. See on põhjustatud sellest, et veebisaidid on kättesaadavad kogu maailmast. Lisaks kasutatakse väga suurt hulka kiiresti arenevaid veebitehnoloogiaid. Nende veebitehnoloogiate ja brauserite standardid on defineeritud lõdvalt [22] – erinevatel inimestel võivad olla erinevad arusaamad. Seetõttu on oluline turvaprobleeme ennetada – üritada leida kitsaskohti enne, kui neid pahalaste poolt kurjalt ära kasutatakse.
Veebisaitide ründamiseks tehakse veebiteenuse vastu päringuid, üritades leida URL-e või lõpp-punkte mis ei käitu nii, nagu ette nähtud. Selle käigus üritatakse koguda veebiteenuse ja seda majutava serveri kohta võimalikult palju informatsiooni mida saab tuleviku rünnakutes ära kasutada. Pärast seda üritatakse erinevate meetoditega sisestada spetsiaalselt selleks otstarbeks meisterdatud päringuid erinevatesse lõpp-punktidesse, et saada mõni komponent veebiserverist

või veebisaidist interpreteerima vastavat päringut käsuna serveri poole peal. Sellist nõrkust saab ära kasutada serverist informatsiooni lekitamiseks, muudatuste tegemiseks või teenusest informatsiooni kustutamiseks [23]. Näiteks SQL keele süstimise rünnak, kus mõnda SQL andmebaasi kasutavasse lõpp-punkti üritatakse sisestada spetsiaalselt konstrueeritud sisendit nii, et SQL andmebaas üritab osa sellest sisendist käivitada.

Lisaks on olemas rünnakuid, mis kasutavad ära veebilehtede kliendi poolset sisu (JavaScript, HTML), et rünnata kolmandaid isikuid. Näiteks ladustatud (inglise keeles *stored*) või peegeldatud (inglise keeles *reflected*) murdskriptimine (inglise keeles *cross site scripting* või *XSS)* [1] [24]. Ladustatud murdskriptimise puhul salvestatakse kliendi poolset pahatahtliku sisu veebisaidi sisse, näiteks andmebaasi, foorumisse või logisse. Kui tavaline kasutaja läheb seda ressurssi vaatama, tõmbab kasutaja brauser pahatahtliku koodi alla ja käivitab selle. Sellise rünnaku võimaldamiseks peab veebisaidi serveri poolne osa salvestama kliendi poolt tulnud informatsiooni nii, et seda ei tehta korralikult ohutuks, ning seda hiljem klientidele tagasi serveerima. Tüüpiliselt kasutatakse seda turvaviga andmete kogumiseks või hajutatud teenusetõkestamise rünnete läbi viimiseks.

Peegeldatud murdskriptimise puhul on tegemist HTTP päringu argumentide nõrkusega, kus serverile saadetud HTTP päringu argumendi sisu tagastatakse kliendile nii, et see kas käivitatakse brauseri poolt, või tagastatakse instruktsioonidena brauserile. Sellisel juhul saab ründaja selle turvanõrkusega URL-i saata kolmandale isikule. Rünnatav isik arvab, et kuna tegemist on usaldatud veebilehega, siis võib seda avada. Kahjuks aga on URL-is juures isevaldne JavaScript, mis teeb misiganes ründaja soovib. Näide õnnestunud peegeldatud murdskriptimise turvaprobleemist Courses veebisaidil on antud loetelus 13.

Selliste olukordade ja rünnakute vältimiseks tuleb kontrollida, ega hallatav veebisait ei luba eelmainitud rünnakuid. Selleks tuleb üritada ise „rünnata“ enda veebilehti ja üritada leida turvavigu. Seda kutsutakse turbetestimiseks (inglise keeles *pentesting* või *penetration testing*) [25].

Järgnevas peatükis kirjeldatakse antud töö raames tehtud turvatestimise tulemusi, kitsaskohti ning võimalikke lahendusi.

5.1 Turbetestid

Turbetestimiseks kasutati vabavaralist tehnoloogiat *OWASP Zap* [26] ja kommertstarkvara *Burp Suite* [27]. Mõlemat kasutati testimiseks, ning ka teise tööriista tulemuste valideerimiseks. Veebisaidi suuruse ja erinevate linkide arvu tõttu keskenduti kahele rünnakutüübile – süstimine (inglise keeles *injection*) ja ebapiisav krüptograafia (TLS).

Pärast rünnaku läbiviimist leidsid mõlemad tarkvarad kaks käsitsi reprodutseeritavat probleemi.

Esiteks, veebisait on vastuvõtlik *ClickJacking* rünnaku vastu [28]. Selle rünnaku käigus laaditakse pahatahtliku brauseris nähtamatu veebilehe peale mingi legitiimne veebileht, näiteks Courses. Ohver arvab, et ta kasutab legitiimset veebilehte, kuid tegelikult mõjutab ta peidetud lehel olevaid toiminguid. Selle nõrkuse saab eemaldada, lisades õige „X-Frame-Options“ küpsise vastusesse, mis ütleb veebibrauseritele kas see veebisait võib kasutada võõraid *iframe* viiteid.

Teine probleem on „Open redirection“, kus spetsiaalselt koostatud päringu abil saab viidata leheküljele, mis siis suunab edasi kolmandale leheküljele vastavalt päringu sisule [29].

```
$ curl -IL https://courses.cs.ut.ee//user/add_bookmark/1065?r
edirect=http%3a%2f%2fa3mtr25yhv1%2fa%3f%2f2017%2feprogalused%2fsp
ring

HTTP/1.1 302 Moved Temporarily
Date: Thu, 17 May 2018 12:51:32 GMT
Expires: Thu, 19 Nov 1981 08:52:00 GMT
Cache-Control: no-store, no-cache, must-revalidate, post-check=0,
pre-check=0
Pragma: no-cache
Location: http://a3mtr25yhv1/a?/2017/eprogalused/spring
Content-Type: text/html; charset=UTF-8
Strict-Transport-Security: max-age=31536000; preload
X-Content-Type-Options: nosniff
X-XSS-Protection: 1; mode=block
Referrer-Policy: no-referrer-when-downgrade
Content-Security-Policy: form-action 'self';
```

Loetelu 13 Näide peegeldatud murdskriptimise "Open redirection" viga ära kasutavast päringust

Selle nõrkuse eemaldamiseks tuleb veebisaidi arendajatel luua nimekiri lubatud suunamise sihtmärkidest ja kirjutada veebilehe suunamise funktsionaalsus ümber nii, et veebileht kontrollib enne suunamist seda lubatud sihtmärkide nimekirja. Kui päringus olev argument pole nimekirjas, siis ei tohi suunamist lubada.

5.2 Shibboleth ja SSO

Hetkeseisuga vastutab Courses veebisaidil autentimise eest PmWiki LDAP moodul, mis edastab autentimispäringud TÜ LDAP serveritele ja vastavalt nende serverite vastusele lubab või keelab kasutajat autentida. Selline lahendus ei sobi suure asutuse kontekstis, kuna eraldiseisvaid teenuseid on raskem kindlustada toore jõu (inglise keeles *bruteforce*) rünnakute vastu, kus üritatakse ära arvata kasutajate parooli. Lisaks, kuna paroolid liiguvad läbi veebisaidi enda loogika, saab veebisaidi koodis oleva turvavea või selle abil üles laetud pahatahtliku koodi abil koguda asutuse kasutajanimesid ja paroole. Kuna asutuste kasutajate paroolid on erinevates teenustes samad, siis ei jää veebisaidi turvaviga lokaliseerituks selle veebisaidi piiresse.

Sellise probleemi lahendamiseks pakub Tartu Ülikooli Infotehnoloogia Osakond SAML 2.0 protokolli kasutavat SimpleSAMLphp tarkvaraga ehitatud *Single Sign On* (SSO) lahendust. SAML protokoll suunab sisselogimiskatsel kasutaja edasi identiteedipakkuja (IdP, inglise keeles *identity provider*) veebisaidini (TÜ korral https://auth.ut.ee), kuid jätab meelde veebiteenuse aadressi, mille pealt edasi suunati. Sellel IdP veebisaidil toimub tegelik sisselogimise ja autentimise katse. Kui see õnnestub, siis suunatakse kasutaja tagasi esialgsesse teenusesse koos kasutaja kohta käivate atribuutidega, nagu näiteks „kasutajanimi“, „roll“, „seosed“. Atribuudid, mis tagastatakse, olenevad SAML konfiguratsioonist. Esialgse veebiteenuseni ei jõua kordagi kasutaja parool, sest selle sisestamine toimub IdP veebisaidil ja seda atribuudina ei tagastata. [30]

Courses veebilehe saab siduda TÜ SSO lahendusega kasutades *Shibboleth* tehnoloogiat. Apache veebiserveri jaoks leidub Shibboleth-i moodul *mod_shib*, millega saab kaitsta veebiressursse, nõudmaks ligipääsu jaoks SSO sessiooni [31]. See nõuab aga Courses veebisaidi arendajate poolt edasiarendust, et kasutaja rollide tuvastamise jaoks kasutataks SSO mitte LDAP vastust. Seda ei saa teha enne, kui on leitud lahendus probleemile kuidas lubada võõrad, ülikooliga mitte seotud kasutajaid, autentima olukorras, kus kasutatakse SSO-d. Väliseid kasutajaid ei eksisteeri Tartu Ülikooli tsentraalsetes kasutajate andmebaasides, mistõttu läbi SSO nad autentida ei saa.

Üheks lahenduseks oleks luua hübriidsüsteem, kus tavalised TÜ kasutajad autentitakse läbi SSO ja teiste kasutajate jaoks on veebisaidil oma andmebaas, kuid see tõstab autentimissüsteemi keerukust märkmisväärselt ja võib paljastada või luua uusi turvaprobleeme.

5.3 ModSecurity

Courses veebisaidi suuruse, keerukuse ja vanuse tõttu on raske selle veebisaiti turvaprobleeme täielikult ennetada. 2004 aasta funktsionaalsuse hulgas võib olla turvavigu mida PmWiki arendajad ei vaevugi tänapäeval parandama, kuna arvatakse, et enam seda ei kasutata. Samas pole kellelgi täielikku ülevaadet tervest Coursesist selle suuruse tõttu.

Seepärast on vaja ka serveri poolele tarkvara, mis kontrolliks HTTP päringute sisu ja semantikat, ning sunniks peale arendajatele häid tavasid, keelates halvad tavad. Sellise ülesandega saavad hakkama kõige paremini veebitulemüürid (inglise keeles *web application firewall* või *WAF*).

WAF-id kontrollivad iga sissetulevat päringut reeglite vastu. Reeglid saadakse turvafirmade töö tulemusel, kus nad kontrollivad õnnestunud rünnakute ründevektoreid ja selle alusel koostavad rünnaku tuvastamise jaoks reegli. Kui WAF-i reegel sobib rünnakuga kokku, siis päring keelatakse.

Tulemüüre on erinevaid, kuid vabavaralisuse ja läbipaistvuse huvides ja kasutatud tarkvaraga arvestades tuleks kasutusele võtta „ModSecurity“, mis on vabavaraline WAF Apache veebiserverile [32]. Lisaks on vaja leida viis kuidas tulemüüri reegleid luua ja hallata. Kuna erinevaid rünnakuid ja nõrkusi on väga suur hulk, siis selle töö tuleks jätta turvafirmade hooleks ja tuleks kasutada vabavaralisi reegleid, näiteks „OWASP ModSecurity Core Rule Set“ [33].

Kahjuks pole sellised tehnoloogiad perfektsed, kuna võib leiduda valepositiivseid ja valenegatiivseid. Nii vana veebisaidi puhul nagu Courses on kindlasti ka vana funktsionaalsust mille peale toetutakse veebisaidi töös, mille WAF ära blokeeriks. Seetõttu tuleb WAF-i integratsioon teha ettevaatlikult, samaaegselt kontrollides millised funktsionaalsused pärast seda enam ei tööta. See vajab head koostööd kasutajate ja arendajatega. See ülesanne jääb arendamiseks ja läbiviimiseks koolivaheajale, kus veebisaidi aktiivne kasutus on väiksem.

6. Seiramine

Seiramiseks nimetatakse süsteemi kvantitatiivsete andmete kogumist, protsessimist, koondamist ja kuvamist [2]. Sellisteks andmeteks võivad olla loendurid süsteemi jõudlusest, päringute arvust või tüübist, vigade hulgast, masinate hetkeolekust jms.
Seiramine on vajalik süsteemi töö automaatseks kontrollimiseks, kasutustrendide ja kasutajakogemuse jälgimiseks ning silumise (inglise keeles *debugging*) lihtsustamiseks. Sellega saab ka saata häireteavitusi vastavalt eeldefineeritud reeglitele. Seiretulemused peaks olema esimene koht kuhu pöörduda probleemide avastamiseks ja tuvastamiseks.

Veebiklastri ja Courses veebisaidi korrektses töös veendumiseks on vaja vastata neljale küsimusele:

- Kas veebiklastri masinad töötavad?
- Kas Courses veebileht vastab päringutele õigesti?
- Kas veebiserverid ja muu tarkvara suudab ära serveerida kõik kasutajad?
- Kas kõigil masinatel on piisavalt ressurssi, et enda tööga hakkama saada?

Lisaks on vaja turbeseiret, et avastada turvaprobleeme serveri, veebiteenuste ja kasutajatega. Selleks tuleb jälgida sisselogimisi, käivitatud protsesse ning serverite poolt algatatud ühendusi. Teadusarvutuste keskusel on kasutusel kaks seiretehnoloogiat. Meetrika kogumiseks ja selle alusel teavitamiseks on kasutusel aegridade (inglise keeles *timeseries*) andmebaas Prometheus [34], ning logide ja struktureerimata andmete kogumiseks ja hoidmiseks on kasutusel Elastic tarkvarakuhi (inglise keeles *Elastic stack*) [35].

6.1 Musta kasti seiramine

Musta kasti seiramine (inglise keeles *blackbox monitoring*) on seiretüüp, kus kontrollitakse teenust nii, nagu väline klient seda näeks ja kasutaks [2]. Prometheus pakub ametlikku musta kasti eksportijat [36], mis võimaldab automaatselt veebiteenuseid pärida ja valideerida vastavalt administraatori sätestatud reeglitele, ning neid tulemusi Prometheus andmebaasi saata.
Courses veebilehe pihta tehakse HTTP päring järgmise reegli alusel iga kahe minuti tagant:

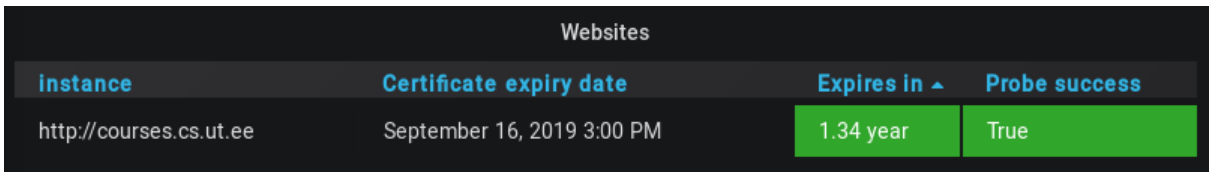
```
https_courses_2xx:
   prober: http
   timeout: 5s
   http:
      method: GET
      preferred_ip_protocol: ip4
      fail_if_not_ssl: true
      fail_if_not_matches_regexp:
      - "Arvutiteaduse instituudi kursused"
```

Loetelu 14 Musta kasti monitooringu reegel Courses tarbeks

Vastava reegli alusel saadakse tekstipõhine vastus, mille alusel saab otsuseid teha. Eriti tähtsad on kolm välja:

- "probe_failed_due_to_regex", mis näitab regex-i kontrolli tulemust, kas HTTP päringu vastuses oli "Arvutiteaduse instituudi kursused" osa olemas. Tavaolukorras on tulemus 0, ehk väär.
- "probe_http_status_code", mis näitab päringu staatuskoodi. Tavaolukorras on tulemus 200, ehk OK [37].
- "probe_success", mis näitab kas HTTP päring oli üleüldiselt õnnestunud. Tavaolukorras on tulemus 1, ehk tõene.

Kui kontrolli tulemused erinevad tavaolukorrast, saadetakse vastavalt probleemi tõsidusele Teadusarvutuste Keskuse süsteemiadministraatoritele kas email või SMS.



Joonis 9 Courses veebilehe automaatne tervisekontroll

6.2 Valge kasti seiramine

Valge kasti seiramine (inglise keeles *white-box monitoring*) on seiretüüp, kus masina sees olev protsess avalikustab või saadab meetrikat ja logisid seireserverile [2]. Jõudluse ja operatsioonisüsteemi meetrika saadetakse Prometheus-sse, ning HTTP päringute ja turvalogid saadetakse Elastic süsteemi.

HTTP päringute logimiseks tuleb kõigepealt HAProxy konfiguratsioonifailis sätestada päringute logimine [38]. Selleks tuleb lisada järgmised read:

```
defaults
      mode http
      log global
      option httplog
```

Loetelu 15 /etc/haproxy/haproxy.cfg faili väljavõte logimise tarbeks

Seejärel hakkab HAProxy logima */var/log/haproxy/haproxy.log* faili. Selle faili sisu saab exportida Elastic tarkvarakuhja ühe komponendi, saatjaga Filebeat, mis on mõeldud tekstipõhiste logide töötlemiseks ja saatmiseks [39]. Pärast sõelumist jõuab logidest tulev informatsioon Elasticsearch andmebaasi. Näide ühe Elasticsearch andmebaasi saadetud HTTP päringu informatsiooni kohta on antud lisas 4. Saadud informatsiooni põhjal saab jälgida pikaajalisi trende serverite töös ja nende jõudluses.

Linux-i auditeerimisraamistiku (inglise keeles *Linux Audit Framework*) sündmuste ja failide tervikluse informatsiooni eksportimiseks [40] pakub Elastic tarkvarakuhi Auditbeat saatjat. Auditbeat-i reeglid töötavad samamoodi varem kirjeldatud Auditd teenuse omadega. Auditbeat-i eelis Auditd ees on automaatne informatsiooni sõelumine ja eksportimine Elasticsearch andmebaasi.

Auditd-d kasutatakse süsteemiadministraatoritel hetkelisete otsuste jaoks vaja mineva spetsiifilise suuremahulise informatsiooni saamiseks, näiteks sissetulevad võrguühendused või ebaõnnestunud sisenemikatsed. Nende sündmuse saatmine Elasticsearch-i hõivaks suure koguse ressurssi nende sündmuste tiheduse tõttu.

Auditbeat-i kasutamise puhul ei pea administraator käsitsi haldama sõelumise reegleid, mis pakub tulevikukindlust ja lihtsamat uuendamist. Kuna selle kasutamine tähendab informatsiooni saatmist üle võrgu, siis on mõistlik masinates jooksutada nii Auditbeat-i kui ka

Auditd-d. Auditbeat jälgimas potentsiaalseid turvaprobleeme mille abil saab koheselt teavitada või tulevikus leitud trendide alusel otsuseid teha. Lisaks aitab selline informatsioon turvaauditel raporteid koostada. Auditbeat-i konfiguratsioonifail on lisas 5.

Auditbeat-iga jälgitakse masinates järgmisi tegevusi:

- Õnnestunud autentimiskatseid ja nende meetodeid
- Konfiguratsioonifailide muutusi
- Autoriseerimata käskude käivitusi
- Väljaminevaid ühendusi
- Serverite käivitudes Linuxi võrgusoklite sidumisi
- Identiteedimuudatusi

Kokkuvõte

Käesoleva töö eesmärk oli uuendada Tartu Ülikooli Arvutiteaduse Instituudi veebisaidi Courses serverid ning leida ja dokumenteerida esinevad turvaprobleemid. Serverite uuendamise tarbeks oli vaja uuendada ka masinate operatsioonisüsteeme. Nõuetest lähtuvalt otsustati alustada algusest ja seada üles uus klaster.

Töö käigus valmis turvaline koormusjaotusega veebiklaster Courses veebisaidi jaoks. Klaster koosneb neljast masinast – kahest proksiserveri ja kahest veebiserveri masinast. Proksiserverite tasemel leiti lahendus kõrgkäideldavuse probleemile kasutades järjestikplaanur DNS süsteemi, virtuaalse IP-aadressi liigutamist kahe masina vahel ning veebiserverite tervise kontrolli. Veebisaitide liikluse analüüsimiseks seati üles seire, mille tulemused saadetakse Teadusarvutuste keskuse Elastic süsteemi.

Veebiserverite tasemel lahendati veebisaidi stabiilsusprobleemid, sidudes teenused omavahel kokku sama masina piires, ning turvaprobleemid kasutades SELinux, PHP-FPM ja auditeerimise tehnoloogiaid, nagu Auditd ja Auditbeat.

Lõpptulemusena valmis klaster, mis suudab veakohad isoleerida ja seejärel töövõime taastada, juhul kui mõni veebiklastri masinatest või teenustest lõpetab töö vea tõttu. See toimib piisavalt hästi, et enamustel kordadel jääb see kasutajale märkamatuks.

Lisaks dokumenteeriti töö raames veebiklastri üles seadmine, kasutajate lisamine, konfiguratsioonihaldus kasutades tehnoloogiat Ansible. Samuti tuvastati ja dokumenteeriti veebiklastri ja Courses teenuse turvaprobleemid. Turvaprobleemide leidmiseks kasutati turbetestimist selleks mõeldud tööriistadega. Tulemused koondati ja nendele pakuti lahendused. Dokumenteeriti ära ka probleemid millele kohest lahendust ei ole. Nende probleemide lahendamine peaks olema esimene fookus valminud süsteemi edasiarendamisel.

Testimise käigus jõuti järeldusele, et veebiklaster vastab kõigile seatud nõuetele. Samas aga leidub ka kitsaskohti, mille lahendamine tõstaks pikemas perspektiivis teenuse töö kvaliteeti ja turvalisust.

Viited

[1] H. Vallaste, „Vallaste e-teatmik,“ [Võrgumaterjal]. Saadaval: http://vallaste.ee/. [Kasutatud 12 mai 2018].

[2] C. a. P. J. a. M. N. R. Jones, Site Reliability Engineering: How Google Runs Production Systems, O'Reilly Media, Inc., 2016.

[3] „Ansible,“ [Võrgumaterjal]. Saadaval: https://github.com/ansible/ansible. [Kasutatud 12 mai 2018].

[4] „YAML,“ [Võrgumaterjal]. Saadaval: http://yaml.org/. [Kasutatud 12 mai 2018].

[5] D. a. B. L. a. C. P. a. D. W. a. D. S. a. F. C. H. a. S. A. a. W. O. a. R. I. Quintero, IBM Spectrum Scale (formerly{GPFS), IBM Redbook, 2015. [Kasutatud 12 mai 2018].

[6] „RFC 2616,“ World Wide Web Consortium, [Võrgumaterjal]. Saadaval: https://www.w3.org/Protocols/rfc2616/rfc2616-sec10.html. [Kasutatud 12 mai 2018].

[7] „Courses arhiivid,“ [Võrgumaterjal]. Saadaval: https://courses.cs.ut.ee/courses/old. [Kasutatud 12 mai 2018].

[8] „CentOS Linux 5 EOL,“ [Võrgumaterjal]. Saadaval: https://lists.centos.org/pipermail/centos-announce/2017-April/022350.html. [Kasutatud 12 mai 2018].

[9] C. N. B. U. I. Pierluigi Stella, „Why You Need a Web Application Firewall WAF,“ [Võrgumaterjal]. Saadaval: http://www.cloudsecurityresource.com/topics/cloud-security/articles/423953-why-need-web-application-firewall-waf.htm. [Kasutatud 12 mai 2018].

[10] „PHP Supported Versions,“ [Võrgumaterjal]. Saadaval: http://php.net/supported-versions.php. [Kasutatud 5 mai 2018].

[11] M. Kreinin, „Riigi Infosüsteemide Amet,“ 2006. [Võrgumaterjal]. Saadaval: https://www.ria.ee/public/publikatsioonid/Mittefunk_nouded.doc. [Kasutatud 7 mai 2018].

[12] „Ansible Documentation - Modules,“ [Võrgumaterjal]. Saadaval: http://docs.ansible.com/ansible/latest/modules/modules_by_category.html. [Kasutatud 17 mai 2018].

[13] K. Gupta, „https://www.freelancinggig.com/blog/2017/04/26/haproxy-vs-nginx-software-load-balancer-better/,“ [Võrgumaterjal]. Saadaval: HAProxy vs Nginx: Which Software Load Balancer is Better?. [Kasutatud 12 mai 2018].

[14] „NGINX Plus,“ NGINX, [Võrgumaterjal]. Saadaval: https://www.nginx.com/products/nginx/. [Kasutatud 12 mai 2018].

[15] „Resource Agents,“ [Võrgumaterjal]. Saadaval: https://github.com/ClusterLabs/resource-agents. [Kasutatud 12 mai 2018].

[16] „Apache HTTP Server Project,“ [Võrgumaterjal]. Saadaval: https://httpd.apache.org/. [Kasutatud 12 mai 2018].

[17] T.-T. a. Q. X. a. L. Z.-H. Cai, „The Improvement of a Data Cached Strategy Based on Eventual Consistency Theory,“ %1 *2016 International Conference on Information System and Artificial Intelligence ISAI*, 2016.

[18] „RFC 1700 - Assigned Number,“ The Internet Engineering Task Force archive, [Võrgumaterjal]. Saadaval: https://tools.ietf.org/html/rfc1700. [Kasutatud 12 mai 2018].

[19] „SELinux - Working with {SELinux} boolean,“ Red Hat, [Võrgumaterjal]. Saadaval: https://access.redhat.com/documentation/en-us/red_hat_enterprise_linux/6/html/security-enhanced_linux/sect-security-enhanced_linux-working_with_selinux-booleans. [Kasutatud 12 mai 2018].

[20] „SELinux/Tutorials/The purpose of SELinux roles,“ Gentoo Linux, [Võrgumaterjal]. Saadaval: https://wiki.gentoo.org/wiki/SELinux/Tutorials/The_purpose_of_SELinux_roles. [Kasutatud 12 mai 2018].

[21] „Apachebench,“ [Võrgumaterjal]. Saadaval: https://httpd.apache.org/docs/2.4/programs/ab.html. [Kasutatud 12 mai 2018].

[22] „The Webstandards Project Archive,“ [Võrgumaterjal]. Saadaval: https://archive.webstandards.org/edu_faq.html. [Kasutatud 4 mai 2018].

[23] „Web Application Security Testing Cheat Sheet,“ [Võrgumaterjal]. Saadaval: https://www.owasp.org/index.php/Web_Application_Security_Testing_Cheat_Sheet. [Kasutatud 5 mai 2018].

[24] „Cross Site Request Forgery,“ About The Open Web Application Security Project Foundation, [Võrgumaterjal]. Saadaval: https://www.owasp.org/index.php/Cross-Site_Request_Forgery_(CSRF). [Kasutatud 3 mai 2018].

[25] „Pentesting: a highly valuable tool for your company,“ Panda Security S.L., [Võrgumaterjal]. Saadaval: https://www.pandasecurity.com/mediacenter/security/pentesting-tool-company/. [Kasutatud 17 mai 2018].

[26] „OWASP Zed Attack Proxy Project,“ The Open Web Application Security Project Foundation, [Võrgumaterjal]. Saadaval: https://www.owasp.org/index.php/OWASP_Zed_Attack_Proxy_Project. [Kasutatud 1 mai 2018].

[27] „Burp Suite Editions,“ Portswigger Ltd, [Võrgumaterjal]. Saadaval: https://portswigger.net/burp. [Kasutatud 1 mai 2018].

[28] „Frameable response (potential ClickJacking),“ [Võrgumaterjal]. Saadaval: https://portswigger.net/kb/issues/005009a0_frameable-response-potential-clickjacking. [Kasutatud 5 mai 2018].

[29] „Open redirection (reflected),“ Portswigger Ltd, [Võrgumaterjal]. Saadaval: https://portswigger.net/kb/issues/00500100_open-redirection-reflected. [Kasutatud 17 mai 2018].

[30] „About SAML,“ OASIS, [Võrgumaterjal]. Saadaval: http://saml.xml.org/about-saml. [Kasutatud 17 mai 2018].

[31] „Shibboleth Wiki,“ [Võrgumaterjal]. Saadaval: https://wiki.shibboleth.net/confluence/display/SHIB2/NativeSPApacheConfig. [Kasutatud 17 mai 2018].

[32] „ModSecurity - About,“ Truswave SpiderLabs, [Võrgumaterjal]. Saadaval: https://www.modsecurity.org/about.html. [Kasutatud 13 mai 2018].

[33] „OWASP ModSecurity Core Rule Set Project,“ [Võrgumaterjal]. Saadaval: https://www.owasp.org/index.php/Category:OWASP_ModSecurity_Core_Rule_Set_Project. [Kasutatud 6 mai 2018].

[34] „Prometheus,“ [Võrgumaterjal]. Saadaval: https://prometheus.io/. [Kasutatud 12 mai 2018].

[35] „ELK Stack: Elastic, Logstash, Kibana,“ [Võrgumaterjal]. Saadaval: https://www.elastic.co/elk-stack. [Kasutatud 12 mai 2018].

[36] „Blackbox Exporter,“ [Võrgumaterjal]. Saadaval: https://github.com/prometheus/blackbox_exporter. [Kasutatud 5 mai 2018].

[37] I. A. N. Authority, „Hypertext Transfer Protocol HTTP Status Code Registry,“ [Võrgumaterjal]. Saadaval: https://www.iana.org/assignments/http-status-codes/http-status-codes.xhtml. [Kasutatud 5 mai 2018].

[38] „Our most loved HAProxy Logging Strategies,“ [Võrgumaterjal]. Saadaval: https://logmatic.io/blog/haproxy-log-what-should-i-do-with-my-haproxy-logs/. [Kasutatud 5 mai 2018].

[39] „Filebeat: Lightweight Log Analysis \& Elasticsearch,“ [Võrgumaterjal]. Saadaval: https://www.elastic.co/products/beats/filebeat. [Kasutatud 5 mai 2018].

[40] „Auditbeat: Lightweight Shipper for Audit Data,“ [Võrgumaterjal]. Saadaval: https://www.elastic.co/products/beats/auditbeat. [Kasutatud 12 mai 2018].

[41] „HAProxy - The Reliable, High Performance TCP/HTTP Load Balancer,“ [Võrgumaterjal]. Saadaval: http://www.haproxy.org/. [Kasutatud 5 mai 2018].

Lisad

Lisa 1 Ansible mall PHP-FPM jaoks

```
[{{ username }}] # Kogumi nimi

user = {{ username }} # Linuxi kasutaja, kelle õigustes failid on
group = {{ username }} # Linuxi kasutaja ja grupp on samaväärsed

listen = /var/run/{{ username }}-php{{ php_version | replace('.',
'') }}.sock # Linuxi sokkel mida kasutatakse selle kogumi poole
pöördumiseks
listen.allowed_clients = 127.0.0.1 # Ainult kohaliku masina
protsessid

listen.owner = {{ username }}
listen.group = {{ username }}
listen.mode = 0660 # Luba ainult õigele kasutajale ligipääs

pm = ondemand # Jõudluse sätted
pm.max_children = 50
pm.start_servers = 1
pm.min_spare_servers = 5
pm.max_spare_servers = 35

slowlog = /php/logs/www-slow.log
chroot = /projects/{{ username }}/ # Kasuta iga kasutaja jaoks
chrooti
chdir = /
catch_workers_output = yes

env[TMP] = /tmp
env[TMPDIR] = /tmp
env[TEMP] = /tmp

php_admin_value[mail.log] = /php/logs/maillog
php_admin_value[error_log] = /php/logs/www-error.log
php_admin_flag[log_errors] = on

php_admin_value[doc_root] = /website_root # chroot-itud failide
asukoht

php_value[soap.wsdl_cache_dir] = /php/sdlcache
```

Lisa 2 Ansible mall veebiserveri HTTP konfiguratsiooni jaoks

```
<VirtualHost *:80>
      ServerName {{ domain }}
      ServerAlias www.{{ domain }}
      ServerAdmin root
      DocumentRoot /projects/{{ username }}/website_root
      ErrorLog /projects/{{ username }}/php/logs/{{ username }}-
error_log
      CustomLog /projects/{{ username }}/php/logs/{{ username }}-
access_log common

      <IfModule log_forensic_module>
            ForensicLog /var/log/httpd/forensic/{{ domain }}-
forensic_log
      </IfModule>

      <Directory /projects/{{ username }}/website_root/>
            Options FollowSymLinks MultiViews
            AllowOverride All
            Require all granted
            DirectoryIndex index.php index.html
      </Directory>

      {% if ForceSSL == "true" %}
      RewriteEngine On
      RewriteCond %{HTTPS} off
      RewriteRule ^ https://%{HTTP_HOST}%{REQUEST_URI}
      {% endif %}


      {% if ForceSSL == "false" %}
      RewriteEngine On
      RewriteRule "^/(.*\.php(/.*)?)$"
"unix:/var/run/{{username}}-php{{php_version | replace('.',
'')}}.sock|fcgi://localhost/website_root/$1" [P,NE]
      {% endif %}

</VirtualHost>
```

Lisa 3 Ansible mall veebiserveri HTTPS konfiguratsiooni jaoks

```
<VirtualHost *:443>
      ServerName {{ domain }}
      ServerAlias www.{{ domain }}
      ServerAdmin root
      DocumentRoot /projects/{{ username }}/website_root
      ErrorLog /projects/{{ username }}/php/logs/{{ username }}-
error_log
      CustomLog /projects/{{ username }}/php/logs/{{ username }}-
access_log common

      SSLEngine on
      SSLCertificateFile    /etc/ssl/selfsigned.crt
      SSLCertificateKeyFile   /etc/ssl/selfsinged.key

      <IfModule log_forensic_module>
            ForensicLog /var/log/httpd/forensic/{{ domain }}-
forensic_log
      </IfModule>

      <Directory /projects/{{ username }}/website_root/>
            Options FollowSymLinks MultiViews
            AllowOverride All
            Require all granted
            DirectoryIndex index.php index.html
      </Directory>

      RewriteEngine On
      RewriteRule "^/(.*\.php(/.*)?)$"
"unix:/var/run/{{username}}-php{{php_version | replace('.',
'')}}.sock|fcgi://localhost/website_root/$1" [P,NE]

</VirtualHost>
```

Lisa 4 Näide Elasticsearch-is hoitud dokumendist


```
{
  "_source": {
        "server_name": "web-ssl1",
        "referer": "https://courses.cs.ut.ee/res/css/theme.css",
        "backend_conn": "0",
        "act_conn": "8",
        "time_total_duration": "24",
        "client_port": "59248",
        "http_method": "GET",
        "backend_name": "ssl",
        "time_wait": "0",
        "accept_language": "en-US,en;q=0.5",
        "client_ip": "193.40.36.253",
        "user_agent":  "Mozilla/5.0  (X11;  Fedora;  Linux  x86_64;
rv:59.0) Gecko/20100101 Firefox/59.0",
        "geoip": {
        "timezone": "Europe/Tallinn",
        "country_name": "Estonia",
        "country_code2": "EE",
        "continent_code": "EU",
        "country_code3": "EE",
        "region_name": "Tartu",
        "location": {
          "lon": 26.7361,
          "lat": 58.3661
        }
      },
      "http_status_code": "200",
      "termination_state": "--VN",
      "frontend_conn": "8",
      "server_conn": "1",
      "http_version": "1.1",
      "bytes_write": "222332",
      "time_connect_backend": "1",
      "request_host": "courses.cs.ut.ee",
      "retries": "0",
      "backend_queue": "0",
      "time_request": "9",
      "@timestamp": "2018-05-13T12:08:51.629Z",
      "frontend_name": "main-ssl",
      "server_queue": "0",
      "http_request": "/res/img/ty_head_1100.png",
      "http_status": "HTTP_OK",
      "time_response_backend": "4"
  },
  "fields": {
      "@timestamp": [ "2018-05-13T12:08:51.629Z" ]
  }
}
```

Lisa 5 Auditbeat-i moodulite konfiguratsioon

```
auditbeat.modules:

- module: auditd
  kernel.socket_type: multicast
  audit_rules: |
      -a always,exit -F arch=b32 -S all -F key=32bit-abi

      -a always,exit -F arch=b32 -S execve,execveat -k exec
      -a always,exit -F arch=b64 -S execve,execveat -k exec

      -a always,exit -F arch=b64 -S accept,bind,connect -F
key=external-access

      -w /etc/group -p wa -k identity
      -w /etc/passwd -p wa -k identity
      -w /etc/gshadow -p wa -k identity

      -a always,exit -F arch=b64 -S
open,creat,truncate,ftruncate,openat,open_by_handle_at -F exit=-
EACCES -k access
      -a always,exit -F arch=b64 -S
open,creat,truncate,ftruncate,openat,open_by_handle_at -F exit=-
EPERM -k access

- module: file_integrity
  paths:
  - /bin
  - /usr/bin
  - /sbin
  - /usr/sbin
  - /etc
  recursive: true
```

Lihtlitsents lõputöö reprodutseerimiseks ja lõputöö üldsusele kättesaadavaks tegemiseks

Mina, Sander Kuusemets

1. annan Tartu Ülikoolile tasuta loa (lihtlitsentsi) enda loodud teose

„Courses veebisaidi serveri kaasajastamine ja turvaprobleemide lahendamine“

mille juhendaja on Kristjan Krips

1.1. reprodutseerimiseks säilitamise ja üldsusele kättesaadavaks tegemise eesmärgil, sealhulgas digitaalarhiivi DSpace-is lisamise eesmärgil kuni autoriõiguse kehtivuse tähtaja lõppemiseni;

1.2. üldsusele kättesaadavaks tegemiseks Tartu Ülikooli veebikeskkonna kaudu, sealhulgas digitaalarhiivi DSpace´i kaudu alates **1. juuli 2019** kuni autoriõiguse kehtivuse tähtaja lõppemiseni.

2. olen teadlik, et punktis 1 nimetatud õigused jäävad alles ka autorile.
3. kinnitan, et lihtlitsentsi andmisega ei rikuta teiste isikute intellektuaalomandi ega isikuandmete kaitse seadusest tulenevaid õigusi.

Tartus, 12.05.2018