TARTU ÜLIKOOL
Arvutiteaduse instituut
Informaatika õppekava

**Sten Kaarel Marvet**

# Privaatse lekkevaba keskkonna loomine veebilehtede uurimiseks

**Bakalaureusetöö (9 EAP)**

Juhendaja:
Alo Peets, MSc

Tartu 2025

## Privaatse lekkevaba keskkonna loomine veebilehtede uurimiseks

**Lühikokkuvõte:**

Tänapäeval on kriminaalid liikunud digimaailma ning nende kinnipüüdmiseks on õiguskaitseasutustel ja küberturbe organisatsioonidel vaja erinevaid digikriminalistika tööriistu. Käesolevas töös on loodud üks selline tööriist, mis on mõeldud veebisaitide aedikkäituseks. Tööriista erinevus olemasolevatest alternatiividest on tingitud võrguülesest ligipääsust ning käiduturbe (OPSEC) säilitamise meetoditest. Tingimustele vastava rakenduse loomiseks kasutati konteineriseeritud veebibrausereid, Docker võrkude isoleerimisvõimekusi ja CSP direktiivi. Aediku paigaldamise lihtsustamiseks, kasutati automatiseerimise rakendusi, nagu Ansible, Terraform ja Docker Compose.

**Võtmesõnad:** RBI, Kasm Workspaces, OPSEC, Ansible, Terraform, Docker

**CERCS:** P175 Informaatika, süsteemiteooria

## The creation of a leakfree environment for investigating websites

**Abstract:**

Modern criminals have moved to the digital world. To catch these criminals, law enforcement and other organizations dealing with cybersecurity need digital forensics tools. This thesis proposes one such tool for sandboxing websites. This tool differentiates itself from the alternatives available on the market with its accessibility over the web as well as using novel solutions for OPSEC. Containerized web browsers, Docker networking and CSP were used to satisfy these unique requirements. To ease the deployment process of the sandbox, tools such as Ansible, Terraform and Docker Compose were used.

**Keywords:** RBI, Kasm Workspaces, OPSEC, Ansible, Terraform, Docker

**CERCS:** P175 Informatics, systems theory

# Sisukord

# 1. Sissejuhatus

Tänapäeva maailm liigub aina enam digitaliseerimise suunas ning erandiks ei ole ka kurjategijad. Nende arengul kannul püsimiseks on nii õiguskaitseasutustel kui küberturbeettevõtetel vaja tööriistu, millega oleks võimalik turvaliselt uurida ning kaardistada kurjategijate tegevust.

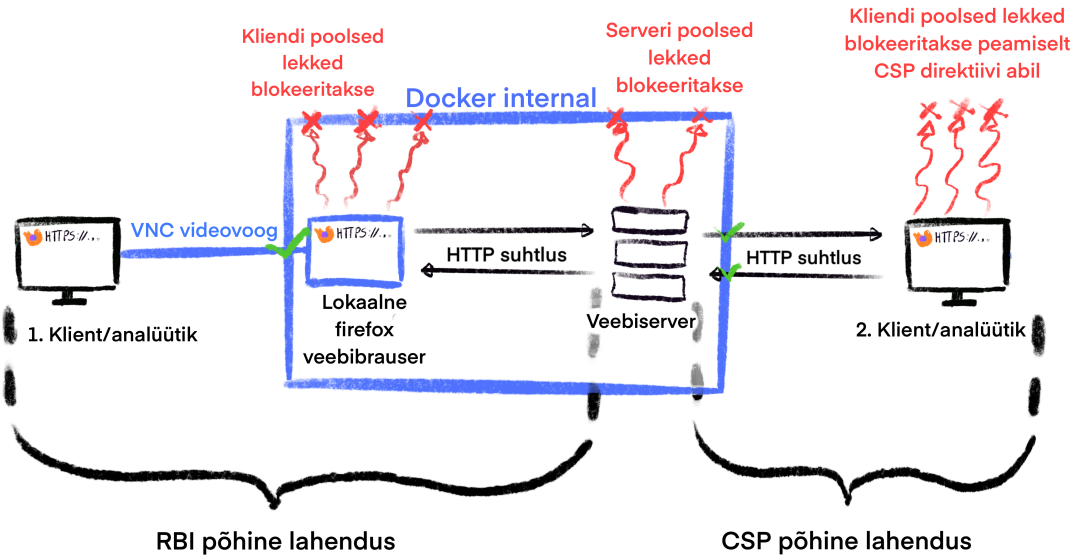
Käesoleva töö eesmärk on luua tööriist, arvestades järgmist olukorda. On olemas kujutletav kriminaalettevõte, mis tegutseb narkootikumide müümisega üle terve Eesti. Narkootikumide müümine toimub veebilehe vahendusel ning selle veebilehe lähtekood ja andmebaas on mingil viisil saanud kättesaadavaks uurijatele või analüütikutele. Veebisaidi turvaline aedikkäitus aitab analüütikutel näha, kuidas narkooperatsioon töötab - kuidas saabuvad tellimused ja maksed ning kuidas hallatakse kullereid ja laovaru. Sellist aedikus käitatavat veebisaiti võib näha demonstratsioonvideos, mille postitas Politsei- ja Piirivalveamet seoses LabHost platvormi sulgemisega [1].

Lahendusena luuakse IaC (*Infrastructure as Code*) tööriist, mis aitaks lekkinud veebisaiti püstitada aedikus järgnevate tingimuste põhjal:

- Kasutatavus - eesmärk on tagada töökeskkond analüütikutele, kes ei pruugi olla eriti kogenud digitaalmaailmas. Seega peab aedikus käitav veebisait olema ligipääsetav läbi brauseri ning turvaliselt kasutatav vähese tehnilise pädevusega isikute poolt. Veebisait peaks samuti olema ligipääsetav üle interneti, et tagada erinevate lokaalsustega inimeste koostööd.
- OPSEC (*Operational Security* ehk käiduturve) - Info aedikkäituse olemasolust ei tohi lekkida veebisaidi algsetele omanikele, välistades nii serveripoolse kommunikatsiooni kui veebibrauseri tehtavate päringute tagajärje.
- Serveriplatvorm ja ligipääs - veebirakenduse kood ei ole turvaline ja võib sisaldada tagauksi, seega on vaja luua isoleeritud keskkond väljaspool organisatsiooni sisevõrku, et mitte ohustada organisatsiooni ennast.
- Autentimine - aedikus veebisait peab olema ligipääsetav ainult selleks volitatud isikutele ning volitamise protsess peab olema võimalikult lihtne.

Aedikkäitus lahendusi on palju, kuid hetkeseisuga ei õnnestunud leida ühtegi, mis vastaks eelnevalt püstitatud nõuetele. Olemasolevad aedikustamise lahendused on mõeldud tehnilisemaks analüüsiks ning ei ole fokuseeritud käiduturvele. Seega loodi suurem osa IaC tööriistast

iseseisvalt ning käiduturve säilitamiseks loodi kaks erinevat võimalust, mille rakendumist on võimalik vastavalt vajadusele kontrollida. Esimene lahendus, mis on ühtlasi ka turvalisem, arendati välja RBI (*Remote Browser Isolation*) baasil ning teine lahendus turvab HTTP suhtlust, rakendades muuhulgas rangeid CSP (*Content Security Policy*) reegleid. Joonisel 1 on näidatud üldine skeem lahendustest.




Joonis 1. Implementatsiooni detailideta skeem CIAB teenustest.

Käesolevas töös loodava rakenduse testimiseks kasutati veebisaiti, mille lähtekoodi ja andmebaasi oli lekitanud üks häkkerite grupp. Käesoleva töö avalikustamise huvides on kõik seda leket identifitseeritavad detailid eemaldatud. Avalikustamise huvides on teatud määral ka muid detaile hägustatud. Tehisintellekt on töös kasutatud ainult CIAB näitelehe kliendiosa loomiseks.

Veebisait koos kaasnevate teenustega (tunnelid, autentimine, turvameetmed) kannavad nime CIAB, ehk *Crime In A Box*. Nimi on inspireeritud Intel NUC sarnastest toodetest, mis on ideaalsed CIAB käitamiseks. CIAB teenuseid tarniv IaC tööriist kannab nime CIAB tarneahel.

Esimene sisupeatükk tööst käsitleb peamisi kasutatud tehnoloogiaid. Järgnevad peatükid kirjeldavad CIAB tarneahela ja CIAB ülesehitust. Muuhulgas on nendes peatükkides lahti seletatud väiksemad tehnoloogiad, mis ei väärinud eraldi alampeatükki. Viimaks on peatükk CIAB tarneahela kasutamisest ning kokkuvõte.

# 2. Kasutatud tehnoloogiad

CIAB ning kaasnev tarneahel on ehitatud kasutades nelja peamist tarkvara: Docker, Cloudflare, Ansible ja Terraform. Lisaks on veel Proxmox VE, kuid see ei ole tegelikult kriitiline osa ning selle asemel on võimalik kasutada suvalist süsteemi, kuhu on paigaldatud Ubuntu LTS (ingl *Long Term Support*) *server* või *desktop* Linuxi distributsioon. Ülejäänud kasutatud tehnoloogiad on väiksemad ning rakenduvad ühel või teisel moel, kasutades eelnevalt mainitud suuremaid tehnoloogiaid.

## 2.1 Docker

Konteiner on grupp protsesse, mis on isoleeritud muudest protsessidest host-operatsioonisüsteemil. Docker on haldusliides, mis võimaldab nii konteinerite kui ka nende vahelise suhtluse defineerimist [2]. Docker koosneb kahest peamisest protsessist, mis tegelevad konteinerite orkestreerimisega: Docker daemon ja *containerd*. Docker daemon on protsess, mis peamiselt haldab Docker objekte ning delegeerib enamuse kohustusi *containerd* protsessile [3]. *containerd* vastutab terve konteineri elutsükli eest, alustades keskkonna loomisest (võrgundus, failisüsteem jne.) ning lõpetades konteineri käivitamise, peatamise ja eemaldamisega [3].

Docker on käesolevas töös kasutusel, peamiselt selle lihtsuse ja populaarsuse tõttu. Docker teeb kasutaja eest konteinerite vahelise võrgunduse ise ära ning tööriistad nagu Docker Compose võimaldavad konteinerite konfiguratsiooni salvestamist. Lisaks on Docker võrdlemisi operatsioonisüsteemi agnostiline. Docker konteinereid on võimalik käitada ka Windowsi ja MacOS masinatel tänu Docker, Inc. loodud kergekaalulisele virtuaalmasinale, mis on osa Docker Desktop komplektist [4].

Dockeri peamine nõrkus on sisemine turvalisus. Dockeri turvalisusprobleeme on vähesel määral käsitletud ja demonstreeritud näiteks Tartu Ülikooli Süsteemihalduse aine praktikumides, kuid konteinerite turvalisus on äärmiselt sügav teema. Dockeri alternatiivina kaaluti Podmani kasutamist, sest erinevalt Dockerist ei kasuta see daemonit, mis on Dockeri peamine turvanõrkus [5, 6]. Podmani ei ole kasusel, sest see on Dockerist märkimisväärselt keerulisem. Käesolevas töös on eeldatatud, et paigaldatavad teenused ei proovi iseseisvalt konteineritest välja murda.

## 2.2 Cloudflare

Cloudflare, Inc. on firma, mis tegeleb kliendi ja serveri vahelise suhtlusega (ingl *client-server communication*). Peamine teenus, mida nad pakuvad, on päringute ja vastuste proksimine läbi nende enda serverite ning võrgu [7]. Peaaegu kõik muud teenused on mõeldud kas toetama proksimist või lisama proksimisele funktsionaalsusi - aadressiresolver võimaldab võrguliikluse suunamist proksidele, CDN (ingl *Content delivery network*) teostab puhverdamist, et kiirendada veebilehtede serveerimist läbi prokside, koormusjaotamine tagab kiiremat pakettide liiklust ja DDoS kaitset, kasutades proksiservereid [7, 8]. Neil on ka teenuseid, mis ei ole seotud proksidega, aga need on pigem erandid. Selles töös on kasutatud Cloudflare autentimisvõimet, aadressiteisendust ja tunneliteenust.

Peamiselt valiti Cloudflare, sest see lihtsustab autentimist ja tagab, et volitamata isikud ei pääseks isegi proksist edasi. Lisaks kasutab RIA (Riigi Infosüsteemide Amet) Cloudflare teenuseid ning on otsutanud, et Cloudflare on piisavalt turvaline [9].

## 2.3 Infrastructure as Code

IaC (ingl *Infrastructure as Code*, ehk taristu koodina) on kirjeldus rakendustele, mis võimaldavad taristu tarnimist ja seadistuste salvestamist. Selliseid tööriistu on mitmeid, kuid käesolevas töös on kasutatud neist kahte.

Ansible on esimene IaC rakendus, mis on kasutatud, et automatiseerida veebisaidile vajaliku taristu tarnet. Ansible tööpõhimõte on võrdlemisi lihtne. Ta kasutab OpenSSH ühendust, et saata sihtsüsteemile moodul (enamasti Python programm), mis teostab seadistuse [10]. Kui moodul on oma ülesande täitnud, kustutatakse see sihtsüsteemist ning saadetakse järgmine moodul [10]. Ansible ülesanne on üldiselt süsteemide seadistamine, aga tehniliselt on sellel sama võimekus, mis oleks sihtsüsteemi käsureal.

Cloudflare toetab ainult rakendusliidese põhist tarnimist ning Ansible pole parim tööriist sellises olukorras. Seetõttu on veel kasutatud Terraform nimelist rakendust, mis on ehitatud kiireks ja asünkroonseks rakendusliidesega suhtluseks [11]. Kiirus pole hetke kontekstis just eriti oluline, aga Cloudflare on võrdlemisi põhjalikult dokumenteerinud, kuidas teostada tüüpilisi seadistusi Terraformi abil.

## 2.4 Proxmox VE

Proxmox VE on Debian Linuxi baasil ehitatud, kohandatud kerneliga virtualisatsiooni haldusliides [12]. See rakendab peamiselt KVM hüperviisorit koos QEMU nimelise riistvara emulaatoriga [12, 13]. KVM, ehk *Kernel-based Virtual Machine* on Linuxi kerneli funktsionaalsus, mis tagab virtuaalmasinatele võimalikult raua-lähedase jõudlusega (ingl *near bare-metal performance*) virtualiseeritud riistvara komponente (CPU, võrguliidesed jne.) [12]. QEMU on riistvara emuleerimise liides, mis võimalusel kasutab KVM võimekust, et tagada paremat jõudlust [13].

Proxmox VE sai käesolevas töös valitud, sest seda on lihtne paigaldada ning Proxmox VE graafiline liides on kasutajasõbralik. VMware Vsphere Hypervisor (tuntud ka VMware ESXi nime all), mis on muidu üks populaarsemaid hüperviisori ja haldusliidese kombinatsioone, jäi hetkel kaalumisest välja, sest 2024. aasta alguses lõpetasid nad tasuta versiooni toe (muuhulgas eemaldasid nad vanade versioonide räsid, millega oleks saanud kontrollida vanade ISO-failide valiidsust) [14]. 2025.aasta 10. aprilli seisuga on avaldatud uus tasuta ESXi versioon, kuid selle kasutamine on endiselt tülikas [15].

Algselt ei olnud plaani teha virtuaalmasinate automaatset tarnimist ning Proxmox valikul seda ei arvestatud. Kui arvestada tarnimisvajadusega, võib parem alternatiiv olla Xen Server hüperviisor, millel on ametlik Terraform tugi [16].

Eelmises peatükis kirjeldati peamiseid tehnoloogiaid ning kuidas nad töötavad. Lisaks selgitati põhjuseid, miks just need tehnoloogiad said valitud, ning millised võivad olla alternatiivid. Järgmine peatükk kirjeldab CIAB tarneahelat ning IaC poolt tarnitavat keskkonda. Tulevastes peatükkides on tihti viidatud spetsiifilistele failidele, mis on leitavad tööga kaasnevast repositooriumist (vt lisa 2).

# 3. CIAB tarneahela ülesehitus

CIAB tarneahel loodi eesmärgiga lihtsustada tervet CIAB algseadistamise ja arendamise protsessi. Erinevalt tüüpilisest seadistusjuhendist tagab tarneahel, et kõik turvalisuse meetmed saavad rakendatud.

Tarneahela kontrollimine käib täielikult läbi Ansible tarkvara. Tarneahela kood asub kaustas *"/CIAB/playbooks/"* ning peamine fail, mis implementeerib kõik võimekused, kannab nime *"master_playbook.yml"*. *"master_playbook.yml"* on Ansible mänguplaan (ingl *playbook*), mis defineerib vajalikud seadistused ja käsud. Siinses kontekstis kutsub ta välja "rolle", mis on modulaarsed seadistuse plokid. Tarneahel koosneb kahest peamisest rollist: "*proxmox*" ja "*webserver*" ning kahest abistavast rollist "*docker*" ja "*terraform*".

Järgmistes peatükkides on kirjas, mida teevad tarneahela rollid, kuid ei ole käsitletud, kuidas käib tarneahela kasutamine, ning mis on vaja eelnevalt ära teha.

## 3.1 Virtuaalmasina tarne kasutades "*proxmox*" rolli

Käesolevas töös on virtuaalmasina operatsioonisüsteemiks valitud Ubuntu Server LTS versiooniga, Linuxi distributsioon, tänu selle populaarsusele ja faktile, et ta toetab *cloud-init* algseadistuspaketti. Lisaks on Ubuntu serverist kogumik pilvekettapilte (ingl *cloud image*), millel toimub aktiivne arendus. Ansible *play*, mis kutsub välja virtuaalmasina loomise rolli, on järgmine:

```
- name: Create VMs on the proxmox host and update the hosts file
 ↪  accordingly
  gather_facts: false
  hosts: proxmox
  tags: cold-boot
  roles:
    - proxmox
```

Sihtsüsteemi kohta info kogumine on välja lülitatud, sest see on hetkel asjatu ajakulu. Ansible *play* on osa mänguplaanist, mis alati teostatakse, ning *tags* argument defineerib sildid, mis tuleb kõikidele *play* ülesannetele (ingl *task*) lisada. *roles* argument otsib üles *"/CIAB/roles/proxmox"* kaustas defineeritud rolli ning impordib kõik ülesanded rollist. *hosts*

argument defineerib grupi või hosti, keda peab seadistama. Algne *"/CIAB/hosts"* fail peaks nägema välja järgmine:

```
localhost ansible_connection=local
[proxmox]
<proxmox hostname> ansible_host=<proxmox ip> ansible_user=root
```

soovi korral võib luua Proxmox hostile uue kasutaja, millel on ainult vajalikud õigused, kuid hetkel ei ole seda teostatud.

"*proxmox*" roll loodi algselt mitme virtuaalmasina loomiseks ning seega on kood mõnevõrra keerulisem kui oleks vaja. Siiski on otsustatud see struktuur paika jätta, et oleks võimalik vajadusel vähese vaevaga lisada virtuaalmasinaid CIAB koosseisu. Seda enam, et kasutades lingitud kloone on süsteemis kasutatud mälu kogus umbes sama. Virtuaalmasinaid on võimalik defineerida *"CIAB/group_vars/proxmox.yml"* failis "*vms*" muutujas. Järgnev on YAML-formaadis kirjeldustega näide:

```
vms:
  template:
    # malli id proxmox hostil (peab olema unikaalne)
    id: 9000
    # malli nimi proxmox hostil (mitmel virtuaalmasinal võib olla sama
    ↪ nimi)
    name: webserver-template
    # sudo õigustega linux vaikekasutaja, mida tulevikus kasutavad
    ↪ Ansible mänguplaanid
    user: ubuntu
  vms:
    # järjend dict objekte
    - id: 9001 # virtuaalmasina id
      # Ansible grupid, kuhu lisatakse virtuaalmasin
      groups:
        - webserver
      # virtuaalmasina nimi
      name: webserver
      # järjend kasutajad, mis luuakse virtuaalmasinale
      users:
        - name: ubuntu
          # grupid kuhu kasutaja lisatakse
```

```
        groups:
          - docker
        # sha-256 räsi paroolist, mille abil määratakse kasutaja
        ↪  parool
        # vaikekasutajale ei tasu parooli määrata, sest ligipääs on
        ↪  tagatud ssh-võtmega
        password: $5$pBs7RJW9Dw2X2Wlx$3fss...
    # staatiline ip mida virtuaalmasin kasutab
    ip: 192.168.1.169
    # virtuaalmasina RAM kogus megabaitides
    memory: 16384
    # virtuaalmasinale määratud tuumade arv
    cores: 8
    # buut ketta suurus
    disk_size: 30G
```

”*proxmox*” roll on teostatud peaaegu täielikult *ansible.builtin.command* mooduliga. Tehniliselt on ka olemas *community.general.proxmox* moodul, mis on spetsiifiliselt loodud Proxmox VE keskkonnas virtuaalmasinate tarneks. Seda moodulit ei kasutatud, sest ta lisab asjatult keerukust. Nimelt on moodul ehitatud ”*Proxmox REST API v2*” rakendusliidesele, ning nõuab, et Ansible mänguplaani käitaval süsteemil oleks installitud Python-põhised ”proxmoxer” ja ”requests” moodulid [17, 18]. Lisaks on proxmoxer moodulil vaja enda spetsiifilisi autentimisandmeid, mis tõstab asjatult tundlike andmete kogust.

Esimene samm ”*proxmox*” rollis on luua SSH võtmepaar ja kopeerida avalik võti Proxmox masina *”~/.ssh/CIAB_authorized_keys”* faili. Võtme nimi on defineeritud *”/CIAB/group_vars/proxmox.yml”* failis muutuja *”ssh_key_name”* nimega.

Virtuaalmasina mall on tegelikult lihtsalt virtuaalmasin, mille käivitamine on keelatud. Virtuaalmasinate loomiseks on proxmox hostil kasutatud QEMU käske. Käsud ise on koostatud Proxmox cloud-init dokumentatsiooni baasil ning käskude kirjeldused on qm käsu dokumentatsiooni baasil [13, 19]. ”*/CIAB/roles/proxmox/create_vm_template_playbook.yml*” teostab järgmised sammud:

1. loob virtuaalmasina käsuga `qm create 9000 --name "template" --agent 1 --net0 virtio,bridge=vmbr0`

(a) `--name "template"` - Defineerib virtuaalmasina nime.

(b) `--agent 1` - Volitab QEMU agendi suhtluse Proxmox VE host masinaga.

(c) –net0 `virtio,bridge=vmbr0` - Defineerib Proxmox hostivõrguliidese, mille kaudu ühendada virtuaalmasin kohalikku võrku.

2. Laeb alla ubuntu pilve-kettapildi.

3. Käsk `qm importdisk 9000 /var/lib/vz/template ...` loob ketta, ubuntu kettapildi baasil ning impordib selle virtuaalmasina ketaste hulka, ilma seda tegelikult kuhugi ühendamata.

4. Teostatakse vaikeseadistus käsuga `qm set 9000 ...`

   (a) `--scsihw virtio-scsi-pci` - Sunnib malli kasutama virtualiseeritud SCSI kontrolleri PCI mudelit [20].

   (b) `--scsi0 local-lvm:vm-9000-disk-0` - Ühendab 3. sammus imporditud ketta, virtio-scsi kontrolleriga.

   (c) `--boot order=scsi0` - Määrab buutseadmeks scsi0 taga oleva seadme.

   (d) `--ide2 local-lvm:cloudinit` - Ühendab cloud-init seadistust sisaldava andmekandja virtuaalmasinaga.

   (e) `--ciuser ubuntu` - Defineerib vaikekasutaja, kellel on muuhulgas sudo õigused.

   (f) `--sshkeys ~/.ssh/CIAB_authorized_keys` - Määrab virtuaalmasina pääsuvõtmeks Ansible poolt loodud SSH-võtme.

   (g) `--cicustom "vendor=local:snippets/vendor.yaml"` - Määrab cloud-init seadistusfaili, mida mallist kopeeritud virtuaalmasinad peaks kasutama. cloud-init ülesanne hetkel on installida QEMU agent ning rebuutida virtuaalmasin. Hetkel on rebuutimine ja QEMU agendi installimine osa sisse jäetud, sest see tagab, et kui virtuaalmasin on kloonitud ja cloud-init skript töö lõpetanud, siis apt on vaba. Vastasel juhul tekib üldiselt olukord, kus apt jääb hanguma. Ilmselt on siin mõistlikum lahendus kui lihtsalt rebuutimine, kuid selle lahenduse leidmine ei olnud hetkel prioriteetne.

5. `qm template 9000` - Määrab loodud virtuaalmasina malliks.

Virtuaalmasina kloonimise etapis on kaks osa. Esimene osa on kloonida virtuaalmasin ning teine on lisada virtuaalmasin *"/CIAB/playbooks/hosts"* faili, et edasised rollid oskaksid seda leida. Vastavalt CIAB algsetele vajadustele on "*proxmox*" roll suuteline looma mitu virtuaalmasinat korraga. Selleks on kasutatud "*proxmox*" rollis,*"clone_and_conf_VMs_playbook.yml"* failis *include_tasks* moodulit koos silmusega. Üldjuhul tuleks sellist itereerimist vältida, sest Ansible ei ole mõeldud olema objekorjenteeritud keel ning abstraktsioonikihtide lisamine kahjustab koodi loetavust. Ansible keerulisema loogika peaks migreerima *jinja2* keeles kirjutatud mallidesse või tuleks Ansible asemel kasutada mõnda teist tehnoloogiat nagu Terraform.

Kloonimine toimub käskudega:

1. **`qm clone 9000 9001 --name "webserver"`** - Kloonib virtuaalmasina, mille id on 9000, ning määrab kloonile id 9001 ja nime "webserver". Loodud kloon on lingitud kloon, mis tähendab, et see kasutab teatud määral malli ketast.
2. **`qm resize 9001 scsi0 30G`** - Suurendab loodud klooni scsi0 ühenduse taga oleva ketta suurust 30 gigabaidini.
3. **`qm set 9001 ...`**
   (a) **`--ipconfig0 ip=192.168.1.169/24,gw=192.168.1.1`** - Seadistab virtuaalmasina staatilise ip ja vaikelüüsi.
   (b) **`--memory 16384`** - Seadistab virtuaalmasina mälu (RAM) koguse.
   (c) **`--cores 4`** - Seadistab virtuaalmasina CPU tuumade koguse.
4. **`qm start 9001`** - Käivitab virtuaalmasina.

Pärast kloonimise lõppu lisatakse virtuaalmasin "*hosts*" faili. Tuvastamaks, millal cloud-init on oma töö lõpetanud pollitakse QEMU agenti, mis sai virtuaalmasinasse installitud. Viimaks sunnitakse Ansible "*hosts*" faili uuendama, et lisatud masinaid saaks seadistada.

Kasutajate ja gruppide loomine käib taaskord *include_tasks* mooduli ja silmusega, kuid seekord on kasutatud moodsamat versiooni Ansible silmustest, sest see võimaldab ise seada muutuja nime ning implementeerida pesastatud silmuse (ingl "*nested loop*"). Taaskord on tegu lahendusega, mille tuleks tegelikult liigutada *jinja2* malli. Hetkel jäeti *jinja2* malli migreerimine tegemata, sest see funktsionaalsust tegelikult ei mõjuta.

## 3.2 CIAB paigaldamine ja algseadistus ”*webserver*” rolliga

”*webserver*” roll kutsutakse *”master_playbook.yml”* failis järgnevalt:

```
- name: Setup the webserver and update it when needed
  hosts: webserver
  roles:
    - webserver
```

”*webserver*” rolli siseselt on kõik ülesanded sildistatud, seega ei ole vaja *tags* argumendiga silte lisada. Erinevalt ”*proxmox*” rollist on ”*webserver*” rollil vaja sihtsüsteemi infot ning faktide kogumine on selle *play* käigus lubatud. Rolli siseselt on *setup* silt kõigil ülesannetel, mis seadistavad keskkonda *”docker-compose.yml”* failis defineeritud konteineritele. *”docker-compose.yml”* sisust tuleb põhjalikumalt juttu ”CIAB ülesehitus” peatükkis. ”*deploy*” silt juurutab (ingl deploy) või uuendab CIAB konteinereid. ”*destroy*” ja ”*destroy-tunnel*” hävitavad Cloudflare tunneli, mis ühendab CIAB konteinerid internetiga.

### 3.2.1 Abistavad rollid ”*terraform*” ja ”*docker*”

”*webserver*” roll kasutab metafaili, et kutsuda välja rollid ”*docker*” ja ”*terraform*”. Erinevalt lihtsamatest tööriistadest, mis leiduvad üldises apt-repositooriumis, on vaja Terraformi ja Dockeri repositooriumid manuaalselt lisada ning seadistada.

Mõlemad rollid laevad alla vastavate repositooriumite gpg-võtmed ning loovad faili, mis kirjeldab apt pakettihaldurile võtmete asukoha. Siiamaani olid rollid peaaegu identsed, aga edasi ilmnevad erinevused. ”*terraform*” roll lõpetab töö pärast terraform paketti paigaldamist. Docker on veidi keerulisem rakendus ning vajab pakette: ”docker-ce”, ”docker-ce-cli”, ”containerd.io”, ”docker-buildx-plugin”ja ”docker-compose-plugin”. ”docker-ce”, ”docker-ce-cli”, ”docker-buildx-plugin” on komponendid, mis üldiselt kannavad Docker Engine nime [21]. ”docker-compose-plugin” on docker konteinerite orkestraator ning ”containerd.io” on pakett, mis haldab konteinerite elutsüklit.

Viimaks määratakse ”docker” rollis konteinerite aadressiresolveriks Cloudflare’i nimeserver ip-addressiga ”1.1.1.1” ning lisatakse Ansible kasutaja (vaikimisi ”ubuntu”) ”docker” gruppi, et ei oleks vaja iga Docker käsu jaoks privileegide taset tõsta.

### 3.2.2 ”*webserver*” rolli ülesehitus

”*webserver*” roll kopeerib *”/CIAB/Docker/”* kausta rsync põhise *ansible.posix.synchronize* mooduliga virtuaalmasina Ansible kasutaja kodukausta. Tüüpiline *ansible.builtin.copy* moodul ei ole optimiseeritud suure hulga failide saatmiseks ning ei sobi, sest *”/CIAB/Docker/”* kaustas on kõik veebisaidi failid (muuhulgas šriftid (ingl font), pildid jne.).

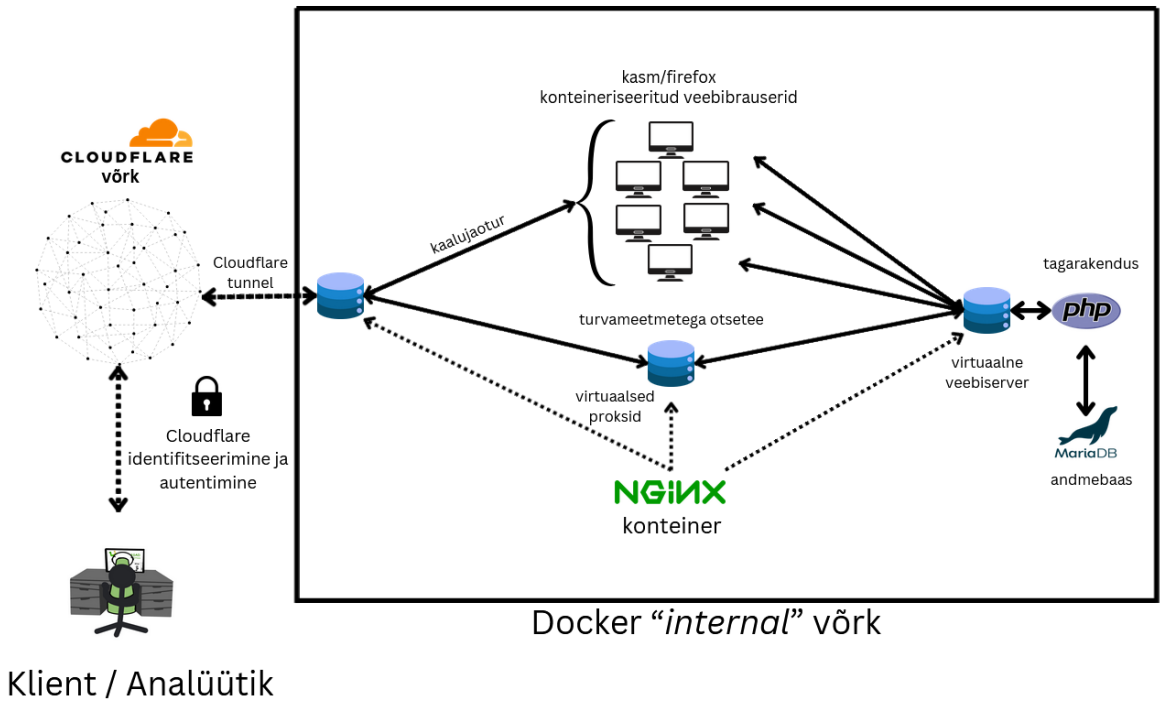
Järgnevalt kasutatakse *community.general.terraform* moodulit koos Terraformi põhise projektiga, et tarnida Cloudflare tunnel. Terraform muutujad sisestatakse läbi *”/CIAB/playbooks/group_vars/webserver.yml”* failis oleva muutuja ”*cloudflare_vars*”. Terraform implementatsiooni spetsiifikat ei ole töös käsitletud, kuid tarnitud teenused on kirjeldatud peatükkis ”Cloudflare tunnel ja pääsurakendus”. Viimaks käivitatakse CIAB konteineriseeritud teenused.

Kui ”*deploy*” silti on kasutatud, siis teostatakse sihtmasinal `docker compose down`, et eemaldada CIAB konteinerid *”/CIAB/Docker/docker-compose.yml”* faili põhjal. Kui konteinerid on eemaldatud, siis uuendatakse *”/CIAB/Docker/"* kausta sisu ning juurutatakse uuesti konteinerid, käsuga `docker compose up -d --build`. Viimane samm on eemaldada kõik sihtmasinasse kopeeritud või loodud failid, kui ”destroy” või ”destroy-ciab” silti on kasutatud. Kusjuures ”destroy” käivitab ka *” master_playbook.yml”* lõpus oleva ”*proxmox*” rolli põhise *play*, mis eemaldab ka loodud virtuaalmasinad.

Käesolev peatükk seletas, kuidas on kasutatud Ansible rolle, et tarnida keskkond, muuhulgas virtuaalmasin, vajalikud pakettid ja CIAB teenused. Järgmises peatükkis on kirjeldatud CIAB teenuseid

# 4. CIAB ülesehitus

CIAB all on mõeldud teenuseid, mille on CIAB tarneahel loonud või tarninud. Näiteks *"docker-compose.yml"* on osa tarneahelast, aga jooksvad konteinerid on CIAB osa. Tehniliselt võiks terve ülesehituse kirjeldus asuda tarneahela all, aga struktuuri huvides on liigutatud n.ö. projekti funktsionaalne tuumik eraldi peatükki. CIAB lihtsustatud ülesehitus on visualiseeritud joonisel 2.




Joonis 2. Lihtsustatud skeem teenuste komplektist.

CIAB teenused ja ressursid on täielikult defineeritud *"/CIAB/Docker/docker-compose.yml"* ning *"/CIAB/roles/webserver/files/Cloudflare-config.tf"* failides. Tehniliselt on võimalik neid kasutada ilma Ansible abita, kuid see vajab natukest ümberseadistamist (peamiselt on Terraformil vaja *tfvars* faili muutujate hoidmiseks, sest hetkel annab muutujad Ansible). Kui edasistes peatükkides räägitakse muutujatest, siis see viitab muutujatele, mis asuvad failis *"/CIAB/playbooks/group_vars/webserver.yml"*.

## 4.1 Cloudflare tunnel ja pääsurakendus

Cloudflare'i tunnel võetakse käesolevas töös kasutusele, sest see võimaldab Dockeri võrkude kasutamist keerulisema tulemüüri asemel. Cloudflare tunnel töötab põhimõtel, et lokaalne tunneli otspunkt võtab ise ühendust Cloudflare võrguga ning edasine suhtlus toimub loodud seanssil. Kodu-keskkonnas võib selline lahendus sobida, aga tasub meeles pidada, et säärane aukude tekitamine tulemüüri võib olla vastuolus suuremate organisatsioonide turvastandarditega.

Cloudflare on äärmiselt kiirete DNS-tsoonitehingutega ning üldiselt agregeeritakse info nimeserveritesse sekunditega, kuid ametlik aeg, mille jooksul need peaks uuenduma, on viis minutit [22]. Testimisel võttis vahetevahel kauem kui viis minutit. Kui aadressiteisendus endiselt ei tööta pärast pikemat ootamist ning konfiguratsioonis pole probleeme, siis võib olla mõistlik konsulteerida Cloudflare klienditoega.

Cloudflare tunnel koosneb kahest osast: lokaalne tunneli konteineriseeritud otspunkt ning tunneli Cloudflare serveripoolne otspunkt koos seadistusega. Lokaalne otspunkt tarnitakse Terraformiga, ning koos otspunkti konteineriga luuakse Docker võrgud, mida CIAB kasutab. Muuhulgas ei tööta Docker Compose ilma Terraform projektita, sest Terraform loob sellele vajalikud võrgud. Võrgud, mis luuakse, kannavad nimesid *public* ja *internal*. Nimedele vastavalt on *public* võrgus ligipääs internetti ja *internal* võrgus on ligipääs ainult teistele konteineritele, mis asuvad *internal* võrgus. Cloudflare tunneli lokaalne otspunkt asub mõlemas võrgus ning võimaldab *internal* võrgus asuva veebiserveri avalikustamist internetti.

Cloudflare poolne seadistus tarnitakse samuti Terraformiga. Cloudflare poolel on CNAME kirje, mis suunab liikluse tunnelile, ning tunnel koos seadistusega. Seadistuses on kirjas teenuse URL lokaalsel otspunktil, vaikimisi *"`https://proxy:8443`"*, ehk NGINX konteineri port 8443.

Cloudflare autentimine toimib kasutades proksil asuvat Pääsurakendust (ingl "*access application*"), millele on rakendatud pääsupoliitika (ingl "*access policy*") ja identiteeditarnija. Identiteeditarnijana on kasutatud ühekordset PIN-koodi, mis saadetakse emailile. Pääsupoliitika lubab emailid, mis on defineeritud "*cloudflare_access_emails*" muutujas. Lisaks lubatakse "*cloudflare_email_domain*" muutujas defineeritud domeeniga emaile, kuid vaikimisi on see välja lülitatud, määrates muutuja väärtuseks ' ' ehk tühi sõne.

## 4.2 CIAB tuumik

CIAB tuumik koosneb konteineritest, mis on defineeritud *"/CIAB/Docker/docker-compose.yml"* failis ning asuvad Dockeri "*internal*" võrgus. CIAB tuumik on ehitatud NGINX konteineri ümber ning kogu liiklus suunatakse läbi selle. Nagu eelnevalt sai mainitud, kuulab NGINX pordil 8443 HTTPS päringuid. Need päringud suunatakse kahte kohta, sõltuvalt URI'st (ingl *uniform resource identifier* ehk ühtne ressursi-identifikaator).

Kui URI väärtus on "/" siis suunatakse liiklus koormusejaoturile, mis omakorda suunab liikluse edasi konteineriseeritud veebibrauserite VNC ühendustele. Veebibrauser avatakse aadressil "http://proxy:9080", kus serveeritakse tegelik veebisait. Teoorias on NGINX seadistuses määratud, et korraga on iga konteineri taha lubatud ainult üks ühendus ning sessioonid peaksid olema jaotatud IP-põhiselt. Selline lahendus ei ole ideaalne, sest samast kohtvõrgust klientidele serveeritakse sama veebibrauser. Cloudflare mingil määral lahendab selle probleemi, sest see teostab enda võrgu siseselt koormusjaotust ning seega on võimalik samast kohtvõrgust ja isegi samast masinast avada mitu sessiooni (NGINX näeb ainult prokside IP-aadresse ja serveerib sessioone nende põhjal). Lahendus töötab kui on vähe kasutajaid, kuid ei ole tegelikult süsteemi haldaja kontrolli all. Seega peaks koormusejaoturi põhise lahenduse asendama mõne robustsema süsteemiga, kuid sellise süsteemi teostamine jäi töö skoobist välja.

Kui URI väärtus on "/direct/" siis suunatakse liiklus proksiserverisse pordil 10080 ning sealt edasi pordil 9080 asuvasse serverisse, kus veebisait tegelikult asub. Selle vahesammu eesmärk on lisada mõned HTTP päised, et vältida veebisaidi olemasolu lekkimist. Lihtsustatud versioon NGINX konfiguratsioonist näeks välja umbes selline:

```
upstream viewer {
    # koormusejaoturi serverid
    ip_hash;
    server viewer1:6901 max_conns=1;
    ...
}

server {
    listen ssl 8443;

    location / {
```

```
    # websocket säted
    ...
    # suunamine koormusejaoturile
    proxy_pass https:/viewer/;
    }

    location /direct/ {
    proxy_pass http://localhost:10080/;
    }

server {
    listen 10080 ;
    CSP;
    JS_filtrid;
    proxy_pass http://localhost:9080;
    }

server {
    listen 9080;
    fastcgi_pass back end;
    }
}
```

### 4.2.1 Lekkevektorid

Lekkeallikaid on CIAB kontekstis tegelikult ainult kaks - server ja veebisaiti külastava kliendi arvuti. Serveri poolel on peamine oht, et veebisaidi tagakomponent proovib iseseisvalt ühenduda välisvõrgus eksisteerivate otspunktidega. Selle keelamine on võrdlemisi lihtne kasutades tulemüüri (käesolevas töös asendatud Docker ”*internal*” võrguga) ning suunates kogu netisuhtluse läbi NGINX veebiserveri.

Peamine raskus on kliendi otspunktil. Peamised lekkevektorid kliendi otspunktil on:

- HTTP päringud - Kui veebileht peaks pärima ressursse mõnelt aadressilt, mis on veebisaidi algse omaniku kontrolli all, siis on võimalik, et päring logitakse ning aediku olemasolu paljastub selle tulemusel.

- Ümbersuunamised - Kui veebisaidi külastaja suunatakse ümber algse omaniku kontrolli all olevale lehele, siis on võimalik, et see külastamine logitakse ning aedikus olev veebisait tuvastatakse selle külastuse kaudu.

Lähtuvalt kaasaegsete brauserite võimekusest on praktiliselt võimatu kaardistada kõiki võimalikke lekevektoreid. Siiski, blokeerides eelnevalt mainitud kaks lekevektorit, on võimalik enamik neist kaudselt peatada. Google Analytics võimaldab külastajate veebisaitide vahelist teed kaardistada ja vaadelda, kuid Google Analytics vajab töötamiseks võimekust teha päringuid [23]. Kui aedikustatud veebisaiti külastatavat kasutajat ei suunata algsele veebisaidile, siis pole algse veebisaidi omanikul põhjust seda kasutajat uurida. Sarnane loogika kehtib ka näiteks brauseri sõrmejäljestamisega (ingl *browser fingerprinting*) [24]. Kui kasutaja ei satu algsele saidile, siis pole põhjust tema sõrmejälge uurida.

### 4.2.2 Konteineriseeritud veebibrauserid ja RBI

Esimeseks lahenduseks lekete peatamiseks on kasutatud RBI ehk *Remote Browser Isolation* laadset võimalust. Suhtlus veebisaidiga toimub Docker ”*internal*” võrgus ning ainuke infovahetus tegeliku kliendiga on VNC põhine videovoog. Kuna keerulisem HTTP suhtlus on lukustatud ”*internal*” võrku, ei ole vaja muretseda, et midagi lekiks.

Turvaline ühendus veebisaidiga on tagatud *internal* võrgus asuvate konteineriseeritud Firefox veebibrauserite kaudu [25]. Need konteinerid on osa suuremast Kasm Workspaces projektist ning kasutavad MIT litsentsi. Lisatud on märkus, et MIT litsents kehtib ainult repositooriumitele, kus hoitakse konteineripakette, kuid nende pakettide sõltuvused ei pruugi olla avatud lähtekoodiga. Põhjalikum koodianalüüs jäi töö skoobist välja, kuid ei tundu, et töös kasutatud *”`kasmweb/firefox:1.16.0`”* konteineripakett kasutaks tehnoloogiaid, mis võiksid litsentsiprobleeme tekitada.

Konteineritesse on võimalik ühenduda üle pordil 6901 oleva KasmVNC ühenduse. KasmVNC peamine eelis on ühenduvus üle brauseri, kasutades veebisoklit (ingl ”*websocket*”). Veebisokkel on jõudluse vaatest vajalik, kuid muudab NGINX abil proksimise äärmiselt tülikaks. Algne plaan oli kasutada URI skeemi, et suunata liiklus staatiliselt erinevatele konteineritele. Kahjuks ei toeta KasmVNC URI põhist suunamist ning staatiline suunamine oleks nõudnud jokker (ingl ”*wildcard*”) alamdomeenide kasutamist, mis testimisel ei teinud eriti head koostööd Cloudflare proksiga.

Koormusejaoturi põhine lõpplahendus ei ole ideaalne ning tuleks asendada Kasm Workspaces orkestraatoriga. Käsolevas töös ei võetud teda kasutusele, sest tasuta *Community Edition* versioon on litsentsiliselt võrdlemisi piirav ning selle teostamine oleks suurendanud töö skoopi märkimisväärselt [26]. Tasuta alternatiiv oleks luua põhjalikum NGINX põhine sessioonihaldus või luua eraldi rakendus, mis haldaks veebibrauserite loomist ja sessioonide serveerimist.

### 4.2.3 HTTP põhine lekete peatamine

Veebisaidi otsese ligipääsu turvamine toimub NGINX serveris, mis kuulab pordil 10080:

```
server {
    listen 10080;
    "object-src 'none'; worker-src 'none'; base-uri 'none'; default-src
    ↪  'self';" always;
    add_header Referrer-Policy "no-referrer";

    # sub_filter_types text/html text/css application/javascript;
    # sub_filter '<domain of the original website>' './direct/';
    # sub_filter 'https://<domain of the original website>' './direct/';
    # sub_filter 'http://<domain of the original website>' './direct/';
    # sub_filter_once off;

      location / {
        js_header_filter main.location_replacer;
        proxy_pass http://localhost:9080;
    }
}
```

Nagu koodinäitest näha, siis esimene samm on lisada CSP ehk *Content-Security-Policy* päis kõikidele vastustele. Edaspidised kirjeldused lähtuvad Mozilla Foundation loodud *mdn webdocs* CSP dokumentatsioonist [27]. CSP on direktiiv, mis ütleb vastuse saanud brauserile, kuhu võib lehe sisu saata päringuid. Näiteks ”style-src ’self”’ tähendab, et CSS laaditabeleid (ingl *stylesheet*) tohib pärida ainult samast allikast, kust päriti HTML. Kasutatud CSP reeglid on:

- ”object-src ’none”’ - Kokkusatumusena on object-src seotud aegunud HTML elementidega ning ”mdn web docs” soovitab selle välja lülitada, kui puudub otsene vajadus. Kui see reegel takistab veebilehe töötamist, siis võib selle eemaldada.

- ”worker-src ’none”’ - Veebitöölised (ingl web worker) käitatakse väljaspool peamise lehe konteksti ning seega ei kehti neile peamise konteksti CSP reeglid [28]. Teoorias peaks siin töötama ka ’self’, sest eraldi päringutega tellitud veebitöölistele lisatakse samuti CSP päis (ingl ”*always*” argument add_header lõpus tagab seda), aga seda ei ole testitud.
- ”base-uri ’none”’ - keelab lehesisese domeeni määramise suhtelistele (ingl *relative*) URL’idele. Taaskord töötaks siin ka ’self’, aga erilist eelist see ei paku.

Referrer-policy ’no-referrer’ tagab, et kui klient satub CIAB domeenis olevalt saidilt tegelikule saidile, siis ei oleks võimalik tuvastada ümbersuunamise allikat. Tehniliselt ei garanteeri see, et infot ei leki, pigem on see järjekordne info sogastamise (ingl *obfuscation*) kiht. Sub filtrid vajavad manuaalset seadistamist ning asendavad veebilehel domeeni suhtelise allikaga. ”`js_header_filter main.location_replacer;`” on javascriptis kirjutatud NGINX njs mooduli põhine filter, mis eemaldab ”*Location*” päises domeeni ning lisab ./direct/ suhtelise URI päise algusesse.

```
function location_replacer(r) {
    if ( typeof r.headersOut['Location'] !== 'undefined'){
        r.headersOut['Location'] = "./direct/" +
        ↪  r.headersOut['Location']
        .replace(/https?(?::\/\/)?(?:[A-z0-9\-]{1,63}\.)+([A-z]{2,63})/,
        ↪  "");
    }
}
```

Domeeniasenduseks loodi IETF (ingl *Internet Engineering Task Force*) RFC (ingl *Request For Comment*) 1035 põhjal regulaaravaldis. IETF RFC 1035 on dokument, millel põhineb domeeninimede reeglistik [29]. Tasub mainida, et seda dokumenti on põhjalikult uuendatud, kuid selle põhjal loodud regulaaravaldis peaks endiselt püüdma kinni suurima osa ASCII-põhistest domeenidest. On teada, et regulaaravaldis ei tööta näiteks IDN (ingl International Domain Name) domeenidega. Selline regulaaravaldis loodi, sest ei õnnestunud leida ühtegi piisavalt põhjaliku kirjeldusega alternatiivi.

HTTP ja filtrite põhise veebibrauseri isolatsiooni peamine nõrkus on tavalised hüperlingid, sest CSP ei mõjuta neid. Üks lahendus, mida prooviti, on kõikide hüperlinkides asuvate domeenide asendamine. Seda teostati taaskord njs põhise filtriga (vt. lisa 1), kuid seekord rakendati filter

vastuse sisule. Kahjuks oli selline lähenemine suhteliselt ebastabiilne, isegi kui rakendada filtrit ainult ”text/html” tüüpi sisu peal. Asendus töötas korrektselt, kuid tegi mingil põhjusel šriftid katki. Rakendades filtrit suurematel failidel, läks failide laadimine äärmiselt aeglaseks.

### 4.2.4 Veebisaidi ühendamine NGINX serveriga

Käesoleva töö peamine eesmärk oli tagada veebisaidi ühenduvus internetiga, seega leiti testimiseks võrdlemisi lihtne veebisait. Tegu oli tegelike kriminaalide poolt loodud veebisaidiga, kuid nagu juba eelnevalt mainitud, on eristatavad detaid välja jäetud. Tegu on PHP-põhise veebisaidiga, mis kasutab MariaDB-andmebaasi. Kõik liiklus Kasm veebibrauseritelt ning turvatud otseteest suunatakse pordil 9080 kuulavale veebiserverile. Seega tuleb kõik veebisaidile vajalik teha selles server-plokis.

```
server {
    listen 9080;
    root /var/html/www/CIAB/;
    index index.php;

    location / {
        try_files $uri $uri/ /index.php?$query_string;
    }
    # Handle PHP files
    location ~ \.php$ {
        fastcgi_pass php-app:9000;
        fastcgi_index index.php;
        fastcgi_param SCRIPT_FILENAME
        ↪  $document_root$fastcgi_script_name;
    }
}
```

NGINX serveri seadistuses tuleb määrata serveeritava sisu failitee ning vaikeleht. Kui päring jõuab *location* direktiivini, siis kõige kõrgem prioriteet on kõige spetsiifilisema regulaaravaldisega *location* direktiivil. Regulaaravaldis ”\.php” on spetsiifilisem kui ”/” ning suunab seega kõik php-lõpulised päringud edasi php-app konteinerile, millel jookseb php-fpm. Kõik staatilised failid käsitleb NGINX ise.

Veebisaidi tagarakendus (ingl *back end*) on üpriski lihtne. Ainuke PHP moodul, mida vaja läheb, on mysqli. Dockerfailis saab ta lisada direktiiviga `RUN docker-php-ext-install`

```
mysqli && docker-php-ext-enable mysqli.
```

Andmebaasi ühendamiseks kontrolliti, mis autentimisandmeid on kasutatud ning loodi nende põhjal andmebaas. Selleks kasutati järgnevaid SQL-päringuid tõmmise (ingl *dump*) faili alguses.

```
CREATE DATABASE db;
CREATE USER 'foo'@'%' IDENTIFIED BY 'asdfghjkl';
GRANT ALL PRIVILEGES ON db.* TO 'foo'@'%';
USE db;
```

Suure tõenäosusega on vaja ka veebisaidi lähtekoodis paar muudatust teha. Näiteleht otsis andmebaasi ”*localhost:3306*” aadressilt, aga CIAB kontekstis on see teises konteineris ning aadress peaks olema ”*db:3306*”.

Käesolevas peatükis kirjeldati CIAB konteinerites pesitsevaid teenuseid ning millised omapärad võivad kaasneda nende teenuste tarneprotsessidega. Lisaks kirjeldati, kuidas peatab CIAB lekkeid, ning kuidas on paigaldatud CIAB komplekti veebisait. Järgmine peatükk käsitleb CIAB tarneahela kasutamist Ansible käskude abil.

# 5. CIAB tarneahela kasutamine

Enne tarneahela kasutamist peab Ansible koodi jooksvale masinale olema paigaldatud vajalikud tarkvarapaketid. Esiteks on vaja installida Ansible ning paar Ansible moodulit käsuga `ansible-galaxy collection install community.crypto community.general ansible.posix`. Kui on soovi kasutada tarneahela võimekust luua virtuaalmasinaid, siis on vaja kõigepealt paigaldada Proxmox VE.

## 5.1 Virtuaalmasina loomine

CIAB oli algselt loodud töötama kasutades Proxmox VE ja üldisemalt virtuaalmasinate funktsionaalsusi. Hiljem õnnestus mahutada kõik CIAB osad Docker konteineritesse ning vajadus Proxmox'i järele kadus. Siiski on soovitatud kasutada virtuaalmasinat, sest see lihtsustab süsteemi seadistamist. Samuti välistab virtuaalmasin olukorra, kus midagi lõhutakse pöördumatult ära, sest vajadusel saab naasta varasemale hetketõmmisele või lihtsalt luua uus virtuaalmasin.

Eeldatud on, et Proxmox VE on juba eelnevalt mõnele füüsilisele serverile installitud. Ansible suhtluse tagamiseks Proxmoxiga on vaja seadistada ssh-võti. Võtme saab luua käsuga `ssh-keygen -f ~/.ssh/CIAB -t rsa -b 4096 -q -N ""` ning kopeerida Proxmox hosti *"~/.ssh/authorized_keys"* faili, käsuga `ssh-copy-id -i ~/.ssh/CIAB.pub root@<proxmox ip>` (ssh võtme nimi võiks olla sama, mis on proxmox.yml failis *ssh-key-name* muutuja väärtus). Järgnevalt on vaja seadistada virtuaalmasina staatiline IP failis *"/CIAB/playbooks/group-vars/proxmox.yml"*. Viimaks on vaja seadistada *"/CIAB/playbooks/hosts"* failis Proxmox VE IP-aadress.

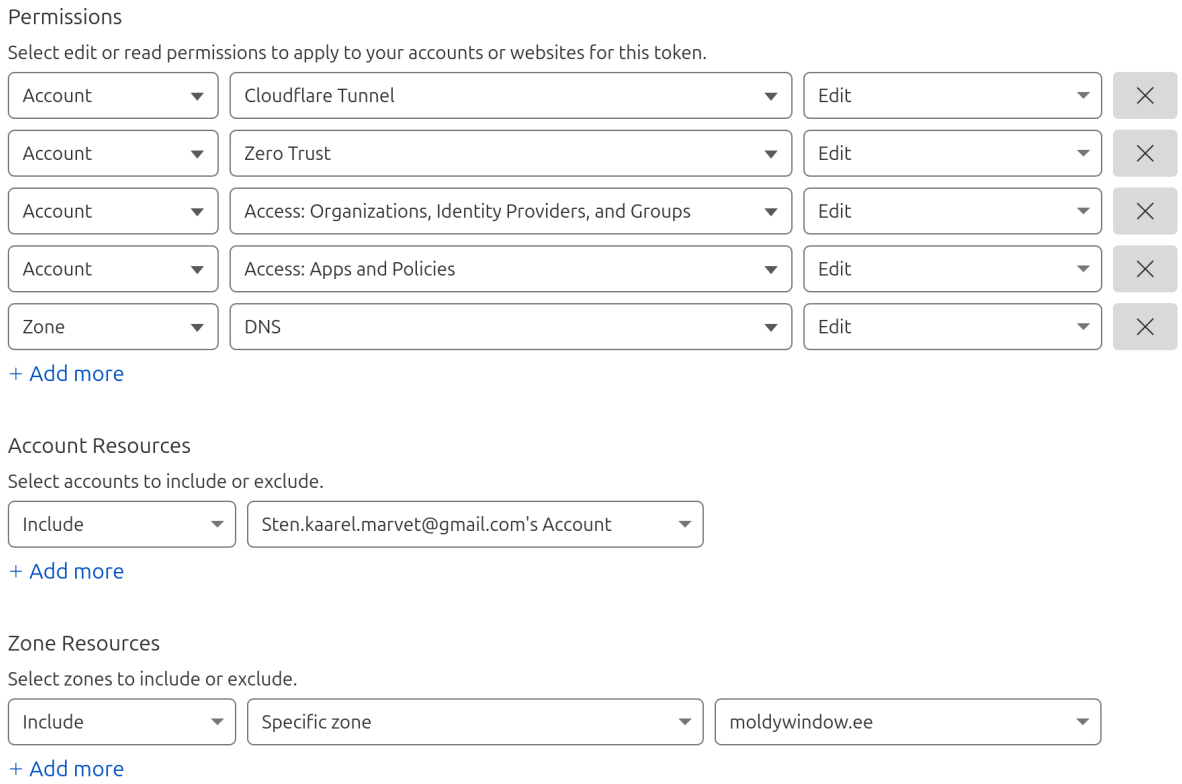
Kui eelhäälestamine tehtud, saab käsuga `ansible-playbook master_playbook.yml --tags cold-boot` luua ja käivitada virtuaalmasina. Muutujas "*ssh_key _name*"defineeritud võti kopeeritakse virtuaalmasinasse ning vajadusel saab seda kasutada SSH-ühenduse loomiseks.

## 5.2 Veebisaidi keskkonna loomine ja CIAB juurutamine

Järgmine samm eeldab, et on olemas Ubuntu-põhine sihtsüsteem ning selle süsteemi IP ja kasutaja andmed on *"/CIAB/playbooks/hosts"* failis. Kui kasutatav sihtsüsteem on loodud tarneahelaga, siis teostati see automaatselt. Lisaks on vaja Cloudflare kasutajat, koos domeeniga. Domeen ei pea olema hangitud Cloudflare'ilt, kuid peab olema migreeritud nende

nimeserveritele.

Keskkonna seadistamiseks on vaja anda tarneahelale piiratud ligipääs Cloudflare rakendusliidesele. Selleks on vaja luua *My Profile* > *API tokens* > *create token* alt rakenduspääse, mille õigused on kirjeldatud joonisel 3.




Joonis 3. rakenduspääse õigused.

Viimaks on vaja ligipääsuandmed salvestada faili *"/CIAB/playbooks/group-vars/webserver.yml"* järgneva koodinäite alusel.

```
cloudflare_vars:
  # domeeni sätted (CIAB seatakse püsti domeenile
  ↪  subdomain.cloudlare_zone)
  cloudflare_zone: "example.com"
  subdomain: "ciab"

  # Domeeni ja kasutaja id, leitav domeeni ülevaate alt
     cloudflare_zone_id: "97asdf798as9f8as97f9sa7f89a7s7f9"
  cloudflare_account_id: "9as9df879s7f89asf9asd7f98sa79df7"
  # Cloudflare kasutaja email. Muuhulgas lisatakse see email
  ↪  automaatselt ligipääsu nimekirja
```

```
cloudflare_email: "example@gmail.com"
# loodud rakenduspääse
cloudflare_token: "LKJADLSJKLJKLASLDJLAJSDKLAJSDLKASJDLJALD"

# ligipääsuga emaili aadressid
cloudflare_access_emails: ["example@gmail.com",
↪  "example.someone.else@ut.ee"]
# See säte lubab emaili domeeni põhiselt lubada ligipääsu (tühisõne
↪  tähendab et sätet ei kasutata)
cloudflare_email_domain: ""
```

Nüüd peaks ka muutujafailid krüpteerima, sest need sisaldavad tundlikke andmeid. Krüpteerimiseks ja dekrüpteerimiseks on võimalik kasutada käsku `ansible-vault <encrypt|decrypt|edit> --vault-id group_vars@prompt ./playbooks/group_vars/*`. Kasutades krüpteeritud faile, peaks tulevikus lisama suvandi `--vault-id group_vars@prompt` kõikidele `ansible-playbook` käskudele. Kahjuks on need andmed sihtmasinal ikkagi paljastatud, sest Terraform ei toeta olekufailide krüpteerimist. Kohalikud muutujafailid peaks krüpteerima, et need ei satuks avatekstis git'i.

Kui ligipääs on tagatud, võib kasutada käsku `ansible-playbook master_playbook.yml --tags setup`, et installida vajalikud tarkvarapaketid ja sätestada Cloudflare tunnel. Kui Ansible on oma töö lõpetanud, siis tasuks kontrollida, kas domeenil on töötav autentimine. Mõne aja jooksul (üldiselt kuni viie minuti jooksul) peaks ilmuma valitud alamdomeenile autentimisvaade.

Enne järgmisele sammule edasi liikumist tasub kontrollida, et pääsurakendus töötab, sest tarneahelal ei ole kontrolli Cloudflare'i teenuste loomise järjekorra üle. On võimalik, et tunnel ja aadressiteisendus töötavad, aga pääsurakendus ei tööta, ning CIAB on ilma autentimiseta ligipääsetav.

Veebiserveri juurutamine (ingl. *deployment*) on vähemalt alguses kõige lihtsam samm, sest näiteveebisaidi juurutamine nõuab lihtsalt käsu `ansible-playbook master_playbook.yml --tags deploy` kasutamist. See kopeerib kõik failid, mis on kaustas *"/CIAB/Docker/"* sihtmasinale, ning ehitab CIAB konteinerid. Näitesaidi asendamist mõne muu saidiga on käsitletud peatükis "Veebisaidi ühendamine NGINX serveriga".

## 5.3 Taristu eemaldamine

Veebisaidi ja kogu kaasneva eemaldamine on automaatne. CIAB eemaldamine on võimalik kasutades käsku `ansible-playbook master_playbook.yml --tags destroy-ciab`. ”destroy-ciab” eemaldab Cloudflare tunneli ja kaasnevad teenused ning eemaldab kõik failid sihtmasinalt. Eemaldamata jääb Docker ja Terraform, sest ei ole võimalik kindel olla, kas need tarnis CIAB tarneahel, või olid need olemas juba varem.

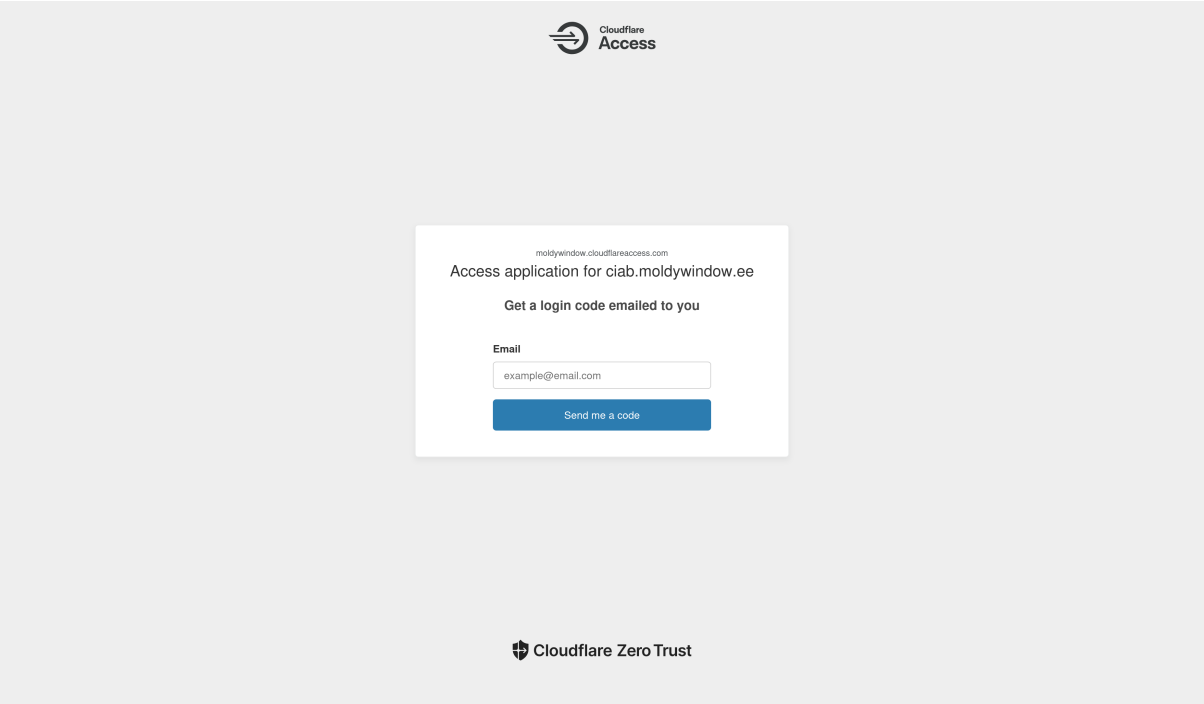
Virtuaalmasina saab eemaldada käsuga `ansible-playbook master_playbook.yml --tags destroy-vms`. Üldiselt on aga soovitatud kasutada destroy silti, mis hävitab nii tunneli kui ka virtuaalmasinad. Kui peaks tekkima olukord, et virtuaalmasin sai hävitatud ilma tunnelit hävitamata, siis on vaja tunnel ja tunneliga kaasnevad osad manuaalselt Cloudflare veebiportaalist kustutada. Peatükis ”Cloudflare tunnel ja pääsurakendus” on kirjeldatud tunneli ülesehitust.

## 5.4 Lihtkasutaja vaade

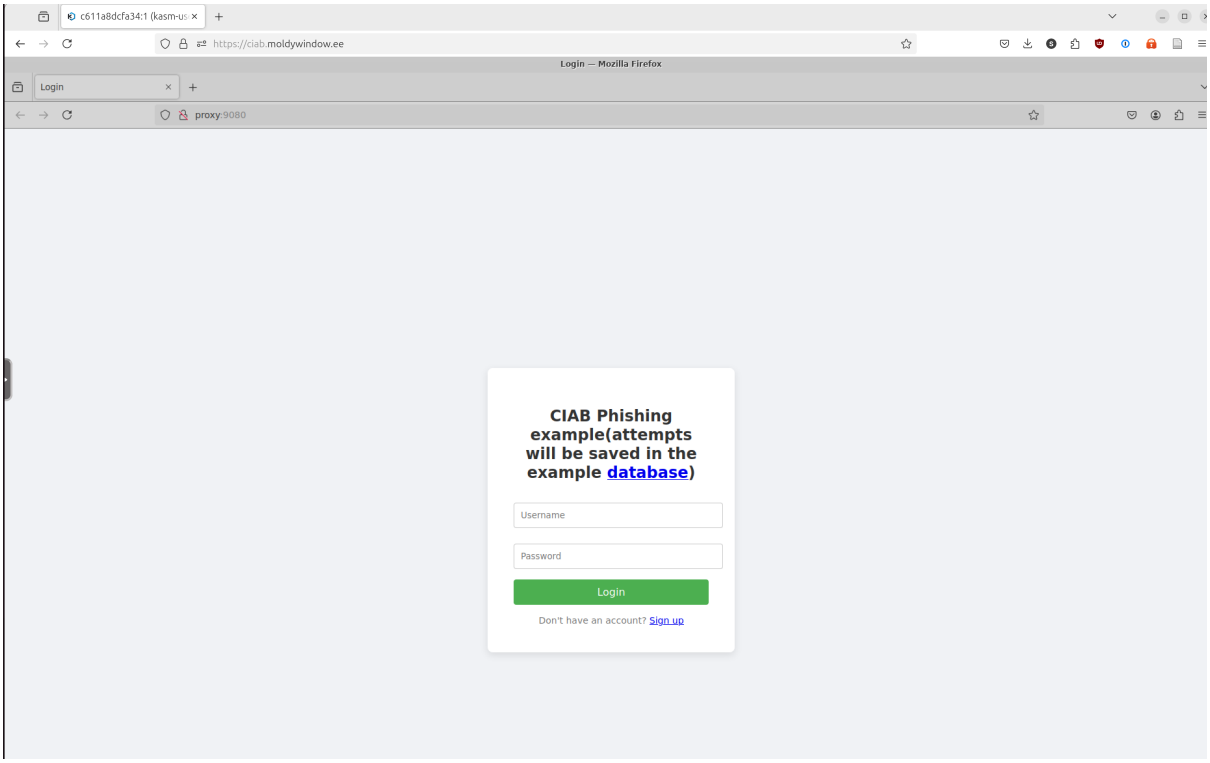
Kui CIAB on korrektselt seadistatud, siis saab tavakasutaja veebisaiti kasutada. Tavakasutaja all mõeldakse näiteks sissejuhatuses kirjeldatud analüütikut. Analüütiku CIAB kasutamise kulg võib välja näha umbes järgmine.

Analüütik sisestab CIAB jaoks seadistatud alamdomeeni endal brauserisse ning selle tulemusena avaneb autentimisvaade, kus küsitakse e-maili (vt joonis 4). Kui analüütik sistestab oma e-maili ning talle on ligipääs lubatud, siis saadetakse talle PIN-kood, mille on vaja autentimisvaates sisestada. Kui PIN on korrektne, siis lastakse analüütik Cloudflare proksist edasi ning ühendatakse ta serveris jooksva veebibrauseriga, kasutades VNC-ühendust (vt joonis 5).

Käesolevas peatükis käsitleti CIAB tarneahela kasutamist ning näidati, milline võib välja näha CIAB kasutamine tavakasutaja vaatest. Järgmine peatükk võtab kokku käesoleva töö sisu, arutab teostatud lahendusi ning pakub välja edasiarendusi.




Joonis 4. Ühekordse PIN-koodi autentimisvaade.




Joonis 5. VNC ühendust kasutav konteineriseeritud veebibrauser.

# 6. Kokkuvõte

Käesolev töö kirjeldab, kuidas on võimalik luua aedik keskkond. Peamiselt on kirjeldatud implementatsiooni detaile ning nendest tulenevaid omapärasid. Metoodika ja tehnoloogiate valiku põhjendamiseks on vastavalt vajadusele kirjeldatud tausta. Võimalusel on pakutud alternatiive, mille kasutamine võib omada eeliseid üle töös kasutatud lahenduste.

Muuhulgas loodi tööriist nimega CIAB tarneahel, mis tarnib isoleeritud keskkonna veebisaidi käitamiseks. Täpsema hinnangu andmiseks CIAB tarneahelale ja CIAB teenustele, võib kasutada sissejuhatuses püstitatud tingimusi.

- Kasutatavus - mõlemad lahendused tagavad ligipääsu üle interneti, kusjuures kasutaja (üldiselt analüütik) ei pea ise midagi seadistama. RBI põhine lahendus võib töötada mõnevõrra ebastabiilselt, kuid väheste kasutajatega ei tohiks probleeme tekkida.
- OPSEC, ehk käiduturve - RBI põhine lahendus tagab käiduturve võrdlemisi kindlalt. CSP põhine lahendus peatab otsesed lekkevektorid, kuid kaasaegsete brauserite keerukuse tõttu ei ole võimalik kindel olla, kas kõik lekkevektorid on blokeeritud.
- Serveriplatvorm ja ligipääs - CIAB on lihtsasti paigaldatav Ubuntu põhisele serverile või Proxmox VE hüperviisorile.
- Autentimine - autentimine on tagatud läbi Cloudflare pääsurakenduse, mida on lihtne kasutada ja seadistada.

Loodud tööriist kasutab vaikimisi RBI lahendust ning seega rahuldab sissejuhatuses püstitatud tingimusi. Lahenduse lähtekood on avalikustatud Gitlab repositooriumis, mille aadressi võib leida lisast 2. Avaldatud repositooriumi litsents ei ole sama, mis on kirjaliku osa lõpus. Soovi korral on võimalik sellega tutvuda repositooriumis.

Üks võimalik edasiarendus on seoses CIAB konteinerite turvalisusega. Hetkel on eeldatud, et CIAB sees käivitatud rakendused ei proovi iseseisvalt välja murda. Kui tagada, et konteinerid on käivitatud turvaliselt, siis oleks CIAB sees võimalik käitada potentsiaalselt ohtlikumaid rakendusi. Teine edasiarendus oleks luua robustsem süsteem konteineriseeritud veebibrauserite loomiseks ja kasutajatele serveerimiseks.

# Viited


[1] LabHost demonstratsioonivideo. Politsei- ja Piirivalveamet. https://www.facebook.com/watch/?v=1658344131575354 (11.05.2025).

[2] What is Docker? | Docker Documentation. https://docs.docker.com/get-started/docker-overview/ (11.05.2025).

[3] Ostrowski S. containerd vs. Docker: Understanding Their Relationship and How They Work Together | Docker. 27. märts 2024. https://www.docker.com/blog/containerd-vs-docker/ (11.05.2025).

[4] Paer-Gotch B. D. S. The Magic Behind the Scenes of Docker Desktop | Docker. 9. september 2021. https://www.docker.com/blog/the-magic-behind-the-scenes-of-docker-desktop/ (11.05.2025).

[5] Security | Docker Documentation. https://docs.docker.com/engine/security/ (11.05.2025).

[6] What is Podman? | Red Hat. https://www.redhat.com/en/topics/containers/what-is-podman (11.05.2025).

[7] What is Cloudflare? | Cloudflare. https://www.cloudflare.com/learning/what-is-cloudflare/ (11.05.2025).

[8] Manish A., Shawn B., Omer Y., Cody D., Alex F. ja Nick W. How Cloudflare auto-mitigated a world record 3.8 Tbps DDoS attack | Cloudflare. 2. oktoober 2024. https://blog.cloudflare.com/how-cloudflare-auto-mitigated-world-record-3-8-tbps-ddos-attack/ (11.05.2025).

[9] RIA kinnitab: Eesti veebilehti kaitsev tarkvaralahendus on turvaline | RIA. https://ria.ee/uudised/ria-kinnitab-eesti-veebilehti-kaitsev-tarkvaralahendus-turvaline (11.05.2025).

[10] Ansible Collaborative - How Ansible Works | Ansible. https://www.redhat.com/en/ansible-collaborative/how-ansible-works (11.05.2025).

[11] What Is Terraform? | IBM. 15. aprill 2025. https://www.ibm.com/think/topics/terraform (11.05.2025).

[12] Features | Proxmox. https://www.proxmox.com/en/products/proxmox-virtual-environment/features (11.05.2025).

[13] qm(1) | Proxmox. https://pve.proxmox.com/pve-docs/qm.1.html (11.05.2025).

[14] End Of General Availability of the free vSphere Hypervisor (ESXi 7.x and 8.x) | Broadcom. https://knowledge.broadcom.com/external/article/345098/end-of-general-availability-of-the-free.html (11.05.2025).

[15] VMware ESXi 8.0 Update 3e Release Notes. https://techdocs.broadcom.com/us/en/vmware-cis/vsphere/vsphere/8-0/release-notes/esxi-update-and-patch-release-notes/vsphere-esxi-80u3e-release-notes.html (11.05.2025).

[16] Automating XenServer with Terraform | XenServer. https://docs.xenserver.com/en-us/xenserver/8/infrastructure-automation/terraform.html (11.05.2025).

[17] community.general.proxmox module – Management of instances in Proxmox VE cluster — Ansible Community Documentation | Proxmox. https://docs.ansible.com/ansible/latest/collections/community/general/proxmox_module.html (11.05.2025).

[18] proxmoxer: Python Wrapper for the Proxmox 2.x API (HTTP and SSH). Versioon 2.2.0. https://proxmoxer.github.io/docs/ (11.05.2025).

[19] Cloud-Init Support - Proxmox VE. https://pve.proxmox.com/wiki/Cloud-Init_Support (11.05.2025).

[20] Laura N. The next-generation storage Interface for the redhat enterprise Linux Kernel Virtual Machine: virtio-scsi | Red Hat. https://wiki.qemu.org/images/c/c2/Virtio-scsi.pdf (11.05.2025).

[21] Install Docker on Ubuntu | Docker. https://docs.docker.com/engine/install/ubuntu/ (11.05.2025).

[22] General Cloudflare | Cloudflare. 14. veebruar 2025. https://developers.cloudflare.com/dns/troubleshooting/faq/.

[23] Qin H., Riehle K. ja Zhao H. Using google analytics to support cybersecurity forensics. *2017 IEEE International Conference on Big Data (Big Data)*. 2017 IEEE International Conference on Big Data (Big Data). Detsember 2017, lk 3831–3834. DOI: 10.1109/BigData.2017.8258385. https://ieeexplore.ieee.org/document/8258385 (11.05.2025).

[24] Mudassar M., Ali M., Ali A., Farid Z., Asif R., Mehmood M. H. ja Salam Mohammed A. An Analysis of Browser and Machine Fingerprinting Techniques. *2023 International Conference on Business Analytics for Technology and Security (ICBATS)*. 2023 International Conference on Business Analytics for Technology and Security (ICBATS). Märts 2023, lk 1–8. DOI: 10.1109/ICBATS57792.2023.10111174. https://ieeexplore.ieee.org/document/10111174 (11.05.2025).

[25] kasmtech/workspaces-images/dockerfile-kasm-firefox at develop | Github. https://github.com/kasmtech/workspaces-images/blob/develop/dockerfile-kasm-firefox (11.05.2025).

[26] Licensing | Kasm. https://kasmweb.com/docs/latest/license.html (11.05.2025).

[27] CSP: default-src - HTTP | MDN. 18. märts 2025. https://developer.mozilla.org/en-US/docs/Web/HTTP/Reference/Headers/Content-Security-Policy/default-src (11.05.2025).

[28] Using Web Workers - Web APIs | MDN. 28. aprill 2025. https://developer.mozilla.org/en-US/docs/Web/API/Web_Workers_API/Using_web_workers (11.05.2025).

[29] Domain names - implementation and specification. Request for Comments RFC 1035. Internet Engineering Task Force, november 1987. DOI: 10.17487/RFC1035. https://datatracker.ietf.org/doc/rfc1035 (11.05.2025).

# Lisad

## Lisa 1. Veebisaidi sisu filter

```
function remove_domains_from_links(r, data, flags){
    let allowed_types = ["text/html; charset=UTF-8"]
    if (allowed_types.includes(r.headersOut['Content-Type'])){
        res += data;
        res = res.replace(
        /https?(?::\/\/)?(?:[A-z0-9\-]{1,63}\.)+([A-z]{2,63})/,
        "");
        if (flags.last) {
            r.sendBuffer(res, flags);
        }
    }else{
        r.sendBuffer(data,flags)
    }
}
```

## Lisa 2. CIAB lähtekood GitLab repositooriumis

https://gitlab.com/sten.kaarel.marvet/ciab

# Litsents

**Lihtlitsents lõputöö reprodutseerimiseks ja üldsusele kättesaadavaks tegemiseks**

Mina, **Sten Kaarel Marvet**,

1. annan Tartu Ülikoolile tasuta loa (lihtlitsentsi) minu loodud teose
   **Privaatse lekkevaba keskkonna loomine veebilehtede uurimiseks**,
   mille juhendaja on **Alo Peets**,
   reprodutseerimiseks eesmärgiga seda säilitada, sealhulgas lisada Tartu Ülikooli digitaalarhiivi kuni autoriõiguse kehtivuse lõppemiseni;

2. annan Tartu Ülikoolile loa teha punktis 1 nimetatud teos üldsusele kättesaadavaks Tartu Ülikooli veebikeskkonna, sealhulgas digitaalarhiivi kaudu Creative Commonsi litsentsiga CC BY NC ND 4.0, mis lubab autorile viidates teost reprodutseerida, levitada ja üldsusele suunata ning keelab luua tuletatud teost ja kasutada teost ärieesmärgil, kuni autoriõiguse kehtivuse lõppemiseni;

3. olen teadlik, et punktides 1 ja 2 nimetatud õigused jäävad alles ka autorile;

4. kinnitan, et lihtlitsentsi andmisega ei riku ma teiste isikute intellektuaalomandi ega isikuandmete kaitse õigusaktidest tulenevaid õigusi.

reprodutseerimiseks eesmärgiga seda säilitada, sealhulgas lisada Tartu Ülikooli digitaalarhiivi kuni autoriõiguse kehtivuse lõppemiseni;

Sten Kaarel Marvet
14.05.2025