TALLINNA TEHNIKAÜLIKOOL
Infotehnoloogia teaduskond

Sander Sapp 178954IADB

Valideerimistarkvara loomine Sercomm IP3442M ruuterile

bakalaureusetöö

Juhendaja: Jaanus Pöial
PhD

Kaasjuhendaja: Innokenti Sobolev
PhD

Tallinn 2024

Autorideklaratsioon

Kinnitan, et olen koostanud antud lõputöö iseseisvalt ning seda ei ole kellegi teise poolt varem kaitsmisele esitatud. Kõik töö koostamisel kasutatud teiste autorite tööd, olulised seisukohad, kirjandusallikatest ja mujalt pärinevad andmed on töös viidatud.

Autor: Sander Sapp

07.11.2023

Annotatsioon

Lõputöö tutvustab süvitsi lähenemist valideerimistarkvara arendamisele IP3442M ruuteri tarbeks, mille omanik on Teleplan Estonia OÜ. Ettevõtte valideerimistarkvara täiustatakse selleks, et olla võimeline läbi viima põhjalikku tootetestimise protsessi. Peamine eesmärk on suurendada ruuteri uuskasutatavust. Projekt on tihedalt seotud kliendi vajadustega ning olemasoleva süsteemi täiustamisega, tõhususe ja töökindluse tagamiseks.

Uurimuse aluseks on analüüs IP3442M ruuteri tehnoloogiatest, uurides selle keerukusi ja funktsionaalsust. See analüüs on oluline, et mõista ruuteri võimeid ja tuvastada kõige tõhusamad valideerimismeetodid. Tarkvaraarendusprotsess on hoolikalt kohandatud sobituma testimiskeskkonna riistvaraspetsifikatsioonidega, tagades töökindla testimise.

Lõputöö pakub ülevaadet võrgutehnoloogia valdkonna professionaalidele ja entusiastidele. See töö annab panuse ruuteri valideerimise praktilisuse valdkonda ja lisab väärtuslikke teadmisi võrguseadmete testimises.

Lõputöö on kirjutatud eesti keeles ning sisaldab teksti 25 leheküljel, 7 peatükki, 19 joonist.

Abstract

Implementing Validation Software for Sercomm IP3442M Router

This thesis presents a comprehensive approach to developing validation software for the IP3442M router, commissioned by Teleplan Estonia OÜ. Company validation software will be improved on to be capable perform in-depth product testing process. The primary goal is to enhance the router's reusability. The project aligns closely with customer needs, emphasizing the refinement of the existing system to ensure efficiency and reliability.

The cornerstone of this research involves an in-depth analysis of the IP3442M router’s technologies, exploring the intricacies of its design and functionality. This exploration is vital in understanding the router's capabilities and identifying the most effective validation methods. The software development process is meticulously tailored to align with the hardware specifications of the testing environment, ensuring robust testing.

Thesis offering a comprehensive resource for professionals and enthusiasts in the field of network technology. This work contributes to the practical domain of router validation and adds valuable insights into the field of network equipment testing.

The thesis is in Estonian language and contains 25 pages of text, 7 chapters, 19 figures.

Lühendite ja mõistete sõnastik

ACS	Automatic configuration server, automaatne konfiguratsiooniserver
AP	Access Point, ligipääsupunkt
CSV	Comma-separated values, komaga eraldatud väärtused
DOM	Document Object Model, dokumendiobjekti mudel
Gbps	Gigabits per second, gigabiti sekundis
GHz	Gigahertz, gigaherts ehk füüsikaline mõõtühik
GPON	Gigabit passive optical network, passivne gigabit optilinevõrk
HDD	Hard disk drive, kõvaketas
HTML	HyperText Markup Language, veebilehe märgendikeel
HTTP	HyperText Transfer Protocol, andmevahetusprotokoll veebis dokumentide vahetamiseks
IEEE	Institute of Electrical and Electronics Engineers, Elektri- ja Elektroonikainseneride Instituut
IoT	Internet of things, asjade internet
ISP	Internet service provider, internetiteenuse pakkuja
JSON	JavaScript Object Notation, andmevahetusvorming
Lambda funktsioon	Anonymous function, anonüümne funktsioon
LAN	Local Area Network, lokaalne võrk
LED	Light-emitting diode, valgusdiood
MoCa	Multimedia over coax, multimeedia üle koaksiaali
RF	Radio frequency, raadiosagedus
RSSI	Received signal strength indicator, vastuvõetud signaali tugevuse indikaator
SSID	Service Set IDentifier, võrgunimi
TCP	Transmission Control Protocol, andmeedastusprotokoll
USB	Universal Serial Bus, universaalne jadasiin
VLAN	Virtual Local Area Network, virtuaalne lokaalne võrk
VoIP	Video over internet protocol, video interneti protokolli kaudu
WAN	Wide Area Network, laivõrk
Wi-Fi/wifi	Wireless fidelity, traadita andmeside standard
WPA	Wireless Protected Access, traadita kaitstud ligipääs
WPS	Wi-Fi Protected Setup, WiFi Kaitstud Seadistus

Sisukord

Jooniste loetelu

1 Sissejuhatus

Telekommunikatsiooni seadmed on tänapäeva elu tavaline osa. Enamik inimestel on internet kodus ning see on loomulik osa. Interneti koju tarnimiseks on vaja, aga ka seadmeid, mis selle töö eest vastutaks. Ruuterid on loodud selleks, et koduseadmeid internetiga ühendada. Tihtilugu renditakse neid seadmeid internetiteenuse pakkuja (edaspidi ISP) käest. Iga kliendi ootus on saada koju töötav seade. Rendilepingu lõppedes saab ISP enda toote tagasi ning soovib veenduda, et järgmisele kliendile toote pakkumisega oleks kõik töökorras. Selle tagamiseks on vaja seade enne järgmisele kliendile tarnimist läbi testida. Väiksemates riikides nagu Eestis on võimalik mahte hoomata üksikisiku tasemel, kus ISP töötaja testib seadme manuaalselt läbi. Arvukama populatsiooniga riikides on mahud suuremad ning seal on automaat testimine ressursi optimeerimise tõttu mõistlikum.

Teleplan Estonia OÜ on innovatsiooni haldusüksus firmas Reconext. Reconext on loodud bränd pärast Clover Technologies ja Teleplan ühinemist. Teenuseid pakutakse Reconext-i brändina. Reconext-i peakontor asub Texases ning kliente on alates Ameerikast kuni Aasia turuni. Ettevõttel on seadete parandamise kogemust 40 aastat. Alates 1983 1 juunist, kui asutati Teleplani televiisorite parandus keskus Saksamaal Frankfurdis.[1]

Käesolevas töös analüüsitakse Teleplan Estonia OÜ olemasolevat lahendust. Autor loob töö lõpuks ettevõtte struktuurile uue IP3442M Sercomm ruuteri testimise lahenduse. Analüüsis selgitatakse välja lõpptulem kaaludes erinevaid tehnoloogiaid ning lahendusi.

Lahendus valmib arvestades ettevõtte Teleplan Estonia OÜ nõudeid ning arvestades kliendi poolt seatud soove ja ülesandeid.

1.1 Taustsüsteem

Ettevõttes on kasutusel testrid, mis võimaldavad toodete testimist. Tegemist on RF signaali blokeerivate seadmetega, mis võimaldab kvaliteetsemat testkeskkonda.

Eelkõige annab see eelise juhtmevabade moodulite testimiseks. Raadiosignaalide tõkestamine on oluline, sest üldjuhul on tehastes kasutusel mitmekümneid või isegi sadu testreid. Seetõttu on oluline, et testimise perioodil ei segaks raadiosignaalid üksteise tegevust.

Võimekus on testida enam levinud sideseadmete tehnoloogiaid. Näiteks: Satelliit, kaabel, VoIP, audio ja video liideste, Wi-Fi, Bluetooth, MoCa, HDD, EHDD ja GPON lahendusi. Testrit iseloomustavad Lisa 2 joonis 11 ja 12.

1.2 Kliendi ootused

Üldjuhul on klientidel väga kindlad ootused ja spetsifikatsioonid, kuidas nende seade peaks käituma enne uuesti turule saatmist. Esineb ka olukordi, kus klient tellib seadme testimise nii, et kindel spetsifikatsioon puudub. Tellitakse antud lahendus Reconext ettevõtte käest. Antud juhul puudus kliendil kindel juhend seadme valideerimiseks. Autoril tuli koostada valideerimis kavand ning seejärel projektijuht kooskõlastas selle kliendiga.

Kavandi koostamisel tuli analüüsida seadme manuaali, mis on väljastatud seadme tootja poolt. Manuaal õnnestus leida internetist, mis sisaldab tehnilisi andmeid toote kohta ning võimaldab mõista, mis elementide töökorda tuleks kontrollida. Kahjuks on tegemist küllaltki lihtsakoelise dokumendiga, kuid annab siiski ülevaate tootest.[2]

Toote tutvustus dokumendile tuginedes on eesmärgiks valideerida toote järgnevad võimekused:

1. Toiteblokk
2. Töötav tarkvara - võimekus töörežiimis olema.
3. Wi-Fi 2.4GHz
4. Wi-Fi 5GHz
5. Ethernet pordid
6. LAN pordid ja WAN port

7. Füüsilised nupud seadmel

8. USB

1.3 Eesmärk

Lõputöö eesmärk on luua ettevõttele vastavalt kliendi soovidele ja ettevõtte võimalustele poolautomaatne lahendus IP3442M Sercomm ruuteri testimiseks.

Täiendus olemasolevale C++ rakendusele, et oleks võimekus valideerida seade nii, et sobiks uuskasutuseks. Seadme testimisel peab arvestama, et seadme tehniline olukord võib olla ettearvamatu. Autor peab looma võimekuse seadmega suhtlemiseks ja testide läbi viimiseks. Test tsükli lõppedes peab süsteem genereerima aruande testitud funktsionaalsustest HTML ja JSON kujul.

2 Suhtlustasandid

Seadmega suhtlemiseks on mitmeid tasandeid. Algselt tuleks seadistada testsüsteem vastavalt, et oleks võimalik esialgne ühendus luua tootega. Selleks on vajalik korrektne võrgukonfiguratsioon seadme ja testri vahel. Samuti on vajalik seadistada ruutingud. Ruutingute jaoks on vaja seadistada operatsioonisüsteemi tasandil ruutingutabelisse liidese alamvõrgu aadress. Lisaks peab olema tabelis ka Mikrotiki ruuting. Selle alusel teab operatsioonisüsteem pakette suunata. Ruutimine toimub operatsioonisüsteemist Mikrotik seadmesse, mille küljes on võrgukaabliga IP3442M ruuter. Mikroitiki ja operatsioonisüsteemi vaheline liiklus on sellega seadistatud. Mikrotiki laetakse konfiguratsioon, mis oskab IP3442M ruuteriga suhelda. Võrgupakettide korrektset liikumist tuleks kontrollida. Selleks võib olla näiteks ping programmi kasutamine. Ping programm on populaarne alamkihi programm, mis on laialt levinud.

2.1 Mikrotiki konfiguratsioon

Mikrotik on seade läbi mille käib testitava tootega suhtlus. Samuti võimaldab Mikrotik testida võrguga seonduvaid liideseid. Mikrotik on alustala seadme testimiseks, sest see loob testrile suhtluskanali tootega. Mikrotik konfiguratsioon kontrollitakse testi alguses ning vajadusel uuendatakse sobivale seadmele disainitud konfiguratsiooniga. Konfiguratsiooni loomise puhul peab arvestama riistvaraga. Testri riistvara võib olla erinev ning seetõttu tuleb skriptis sellega arvestada. Vanemad Mikrotikid toetavad ainult ühte traadita võrgu liidest, mis tähendab, et kahe võrgu testimiseks peab vahepeal ümberlülitus toimuma. See funktsionaalsus on lahendatud C++ rakenduses.

Skript seadistab Mikrotiki algväärtustega. Nii on lihtne ja mugav rakenduses hiljem väärtusi muuta. WiFi turvaprofiil seadistatakse algselt WPA ja WPA2 parooliks *defaultkey*. WPA2 on uuem ja turvalisem versioon[3]. WPA on mõistlik ainult siis kasutada kui toote tarkvara ei võimalda WPA2[4].

Mikrotik on võimeline olema *station* ja *ap-bridge* olekutes. Sercommi ruuter on võrgujagaja seega on tegemist AP seadmega. Järelikult on vaja Mikrotik seadistada klient olekusse, et õigete parameetrite olemasolul loodaks traadita võrk.

WiFi liidese konfigureeritakse vastavalt Mikrotiki mudelile. Võimalusel mõlemad liidesed vastavalt wlan1 5GHz jaoks ja wlan2 2.4GHz jaoks. Samuti seadistatakse antenni signaali tugevus *antenna-gain=3*. Signaali tugevus on oluline, sest liiga tugeva signaali korral ei pea seade WiFi võrguühendust looma. Võimsuse seadistamisel väiksem number tähendab tugevamat signaali. WiFi turvaprofiil väärtustatakse eelnevalt loodud profiiliga.

WAN liidese seadistamiseks kasutatakse ether6 porti Mikrotikis ning vlan-id seadistatakse 50. IP määratakse 192.168.34.1/24. Kladkriipsuga eraldatud 24 viitab võrgumaski kõikidele aadressidele IPv4 võrgu olukorras. Samaväärne võrgumask on 255.255.255.0. WAN liidesele on vaja IP seega määratakse aadresside vahemik(*pool*) 192.168.334.10-192.168.34.100. Tegemist on ainult ühe tootega, siis saaks ka kasutada ainult ühte aadressi. Liidese jaoks tehakse spetsiaalne DHCP server, mis jagab IP vahemiku aadressi liidestele.[5]

DHCP klientideks määratakse kõik liidesed, mida seade hakkab kasutama välja arvatud WAN port. Mikrotikist liides 2, 3, 4, 7 ja traadita võrgu liidesed wlan1 ja wlan2. Mikrotiki seitsmes liides läheb serverisse ega kasuta tavalist ruutingut. Seda liidest kasutatakse kiirustesti läbiviimisel.

Viimaseks seadistatakse tulemüür (Joonis 1). Tulemüüris on vaja ära märkida, mis liideste kaudu andmepaketid liikuda võivad. Tänu sellele konfigureerimisele saavad paketid liikuda mõlemas suunas. Vastasel juhul liiguksid paketid ainult Mikrotik sisse, aga tagasi ei saaks liikuda. Siin deklareeritakse kõik liidesed, mida seade kasutab.

```
/ip firewall nat add action=masquerade chain=srcnat out-interface=ether1
```

Joonis 1. Esimese liidese tulemüüri seadistamine

2.2 Suhtluskanalid SSH ja Telnet

Seadmega suhtlemiseks on vajalik suhtlustasand. Kõige parem ning mugavam oleks otse ühendus seadmesse, kus oleks võimalik ruuteri operatsioonisüsteemi tasemel käsklusi edastada. SSH pordiks on 22 ja Telneti pordiks on 23. Tegemist on

populaarsete võrguühenduskommunikatsiooni loomise vahenditega ning seetõttu on fikseeritud pordid. Teades neid porte saab skaneerida IP3442M ligipääsetavaid porte. Selleks kasutab autor nmap rakendust. Nmap on vabavaraline tarkvara. Käsuna saab kasutada nmap -sn IP aadress. S ja n parameetrid annavad ping skaneerimis võimaluse. Sellisel juhul saadetakse ping pakett populaarsematele portidele. Samuti on võimalik üle kontrollida kõik seadme pordid, aga see on aega nõudvam tegevus. Skaneerides seadet ei tuvastata avatud porti 22 ega 23, mis on ka oodatav tulemus. IP3442M kasutab äri eesmärkide täitmis tarkvara, mis turvakaalutlustel ei luba SSH ega Telnet ühendusi teostada.[6]

2.3 Automaatse konfiguratsiooniserver

ACS on serverrakendus, mis võimaldab juhtida ühendunud seadmeid. Eelkõige on tingimuseks klient seadme TR-069 standardile vastavus. ACS on järjest rohkem populaarsust koguv seadme haldamismetoodikaid. ISP pakub see teenus mugavust, sest nad saavad kodukasutajate seadeid vajadusel juhtida. Eesti turul kasutab sarnast lahendust Telia. Autori projektis ACS kasutus on, aga tülikas ja keerukas, sest see nõuab lisa serveri olemasolu. Samuti oleks vaja seadme tootja poolset lisainfot, kuidas võimalusel selle ühendamine peaks toimuma.[7]

2.4 Arvutitarkvara projekt CURL

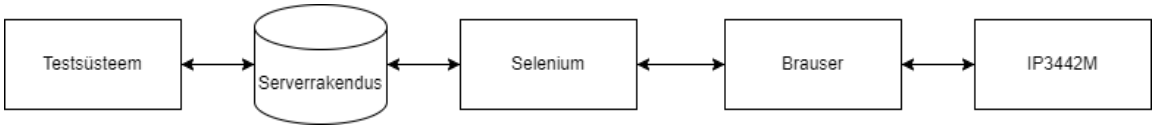
Ainus lähenemisvõimalus siiani seadmele on veebibrauseri kasutus. Järelikult on vaja kasutada HTTP päringuid. CURL on selle jaoks hea konsoolipõhine rakendus. Tegemist on vabavaralise tarkvaraga, mis on laialt levinud. Selle rakenduse abil saab seadmesse sisse logida ja HTML lehekülge kuvada. IP3442M kasutajaliides on Javascripti põhine. CURL nõrkus on see, et sellel rakendusel puudub võimekus Javascripti elemente juhtida.[8]

2.5 Moodul Selenium

Selenium moodul võimaldab juhtida veebilehekülge, mis kasutab Javascript programmeerimiskeelt. Teek on loodud kasutajaliidese testide kirjutamiseks, kuid saab simuleerida tavakasutaja käitumist. Nendel põhjustel on autor valinud Selenium teegi ruuteri juhtimis suhtluskanaliks.

3 Testritarkvara

Rakendus on kirjutatud programmeerimiskeeles C++, selles osas loob autor äriloogika, et seadet oleks võimalik valideerida. Tulenevalt programmeerimise heast tavast ja järgides mustreid on mõistlik jaotada üleüldiselt rakendus kihtideks[9]. Rakenduse kihistumine annab võimaluse paindlikkusele ja paremini loetavusele (Joonis 2).



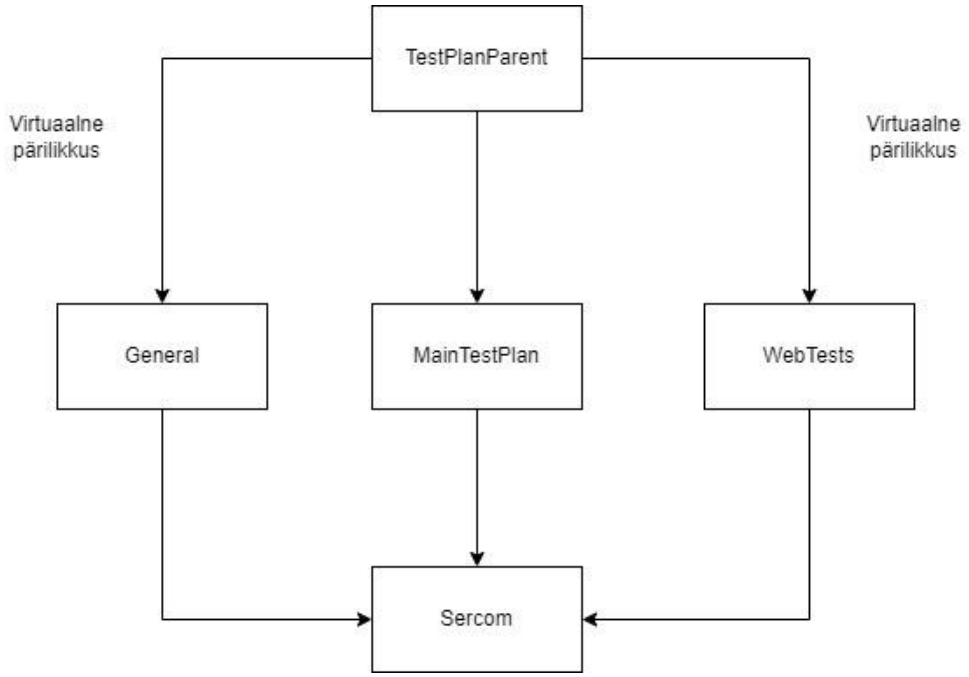
Joonis 2. Tootega suhtlus

C++ äriloogika on jagunenud kahte klassi. WebTests ja Sercom. Sercom klass sai loodud eesmärgiga toote spetsiifiliste tegevuste juhtimiseks ning WebTests klassi eesmärk on üles seadistada Seleniumiga seotud tegevused. Selenium teeki kasutab autor läbi programmeerimiskeele Python. Skripti käivitamine ning juhtkäsud saadetakse C++ rakendusest (Joonis 2). Webtests sai loodud eesmärgiga arvestada ka tulevikus uute toodetega, mida peab läbi Seleniumi rakenduse juhtima.

3.1 Klass WebTests

Tegemist on üldise klassiga, mida peavad saama kasutada kõik testplaani klassid, kus on vaja Selenium ühendust luua. Klassi eesmärk on hoida ühte ühendust Selenium serveriga kogu testi tsükli käigus selleks, et ei peaks WebDriver rakendust korduvalt käivitama Samuti saab päringuid teha ühe sessiooni raames. Klassis luuakse TCP ühendus programmeerimiskeeles Python kirjutatud serveriga. Põhjus miks serveri pool on programmeerimiskeeles Python kirjutatud tuleneb sellest, et see annab rakendusele paindlikkust. Tulenevalt C++ omadustest, et rakendust tuleb kompileerida teeb see versioonide uuendamise tülikamaks. Python rakendust saab käsitleda konfiguratsioonina seadme juhtimiseks.

WebTests klass pärineb virtuaalselt TestPlanParent klassist, sest sellel klassil on kohustus luua testisammude tulemusi. Virtuaalne pärilikkus on oluline, et vältida teemant arhitektuuri teket (Joonis 3).



Joonis 3. Klasside arhitektuur

Esimeseks testisammuks, mis selles klassis rakendub on StartWebService. See on mõistlik esimesena käima panna sellepärast, et see loob TCP ühenduse Python serveriga. Testisamm omab ka parameetrit, mille alusel skript käivitatakse. Lisaks on ka ajalimiit, sest on esinenud Chromium rakenduse aeglast käivitamist. Serveri eduka käivitamise puhul saadetakse esimene test käsk, mis sisaldab parooli sisselogimiseks. Seleniumile käskude saatmiseks kasutatakse SeleniumServer teeki, mis on mõeldud toetama Python rakenduse käivitamist.

Esimeseks testisammuks kui toode on käivitunud soovitakse toote seadistada tehaseseadetesse. See tagab toote etteaimatava käitumise. Selle jaoks on testisamm BootUpAndReset. Antud meetodi päisefailis on lisaks meetodi deklaratsioonile ka lisa informatsioon. Selle informatsiooni pealt genereeritakse rakenduse ehitamise ajal rakendusele juhis parameetrite kohta. Parameetreid on rakendusel võimalik võtta mitmest eri kohast vastavalt parameetri tüübile. Antud juhul on parameetrid ettenähtud testiplaani JSON failist. See tagab mugava ja käepärase seadistamise rakenduse kasutajale. Antud parameetrit saab samuti muuta ka rakenduses testiplaani redaktori kaudu (Joonis 15). BootUpAndReset meetodil on kolm parameetrit. Tuleb määrata

ajalimiit, kui kaua oodatakse seadme töökorra olekuni. Interneti kaablite arvu määramine on oluline seetõttu, et selle alusel saab testri loogika ühendusi luua. Kui esimene kaabel on juhuslikult katki või on seadme poolne viga, siis saab järgmist liidest proovida. Viimaseks parameetriks on viiteaeg. Selle eesmärk on seadistada tihendus, kui tihti kontrollitakse seadme töökorda. Samuti näeme ka lisainformatsioonist, et meil võib meetodi rakendamisel tulla veaolukord. Veaolukord tähendab seda, et test jääb seisma ning ei tagasta kindlat staatust seadme kohta. Sellisel juhul testri kasutaja näeb, et mingil põhjusel test ebaõnnestus ning teab, et seade on endiselt testimata. Antud meetodis saab tulla veaolukord, kui Python serveriga kaob ühendus või ei saada oodatava aja jooksul vastust (Joonis 4).

```
    /*
     * {
     * "Name": "WebTests::BootUpAndReset",
     * "Description": "Check if DUT booted up and try to log in. If login
successful perform automatic reset, if not ask for manual reset",
     * "Pass": "DUT booted up. Reset Successful",
     * "Fail": "DUT failed to boot up. Reset failed.",
     * "Error": "Failed to send command to server.",
     * "Parameters": [
     *   "int BootUpTimeout: timeout in seconds for boot up check for single
Ethernet interface",
     *   "int ResetDelay: delay in seconds after factory reset inititalized",
     *   "int EthAmount: amount of eth ports"
     * ]
     * }
     */
    void T_BootUpAndReset(const FunctionResult& fResult, const
QList<TestParameterItem>& parameters);
```

Joonis 4. Meetodi deklaratsioon päisefailis

Esimese asjana proovib tester läbi Selenium rakenduse saavutada algse seisu. Selleks saadetakse Selenium serverile CheckBootUp käsk. Sellele käsule on pandud küllaltki suur ajalimiit 50 sekundit. Põhjus tuleneb sellest, et antud klass peab suutma hallata erinevaid tooteid ning mõned tooted võivad väga aeglased olla. Lisaks on tegemist ajalimiidiga, mis tähendab, et normaalses olukorras tagastatakse serveri poolt tulemus kiiremini. Käsu vastuseks oodatakse null, üks või kaks numbriga tagastuskoodi. Kui on tulemuseks kood kaks, siis see tähendab, et seade on juba mingil põhjusel sisse logitud ning saab jätkata järgmiste tegevustega. Kui on tulemuseks on üks, siis järelikult ei ole õnnestunud veel oodatavad aadressi kätte saada ning peab jätkama proovimist. Kui tagastuskoodiks on null, siis järelikult oleme saavutanud algse aadressi ning saame

proovida sisselogimist. Kui sisselogimine ebaõnnestub, siis saame järeldada, et tootel on teadmatu parool ning vajame operaatori abi. Operaatori kutsumine toimub ainult juhul kui tester ei ole tsükkel režiimis. Operaatorile kuvatakse instruktaaž Localaizer teeki kasutades. Teek võimaldab saata juhend vastavalt konfigureeritud keelele. Põhjus miks operaatorit kohe alguses ei kutsuta on selles, et inimressurss on kallis ja piiratud. Äriliselt on mõistlikum proovida esialgu testril endal.

Kui on õnnestunud seadmesse sisse logida. Saame jätkata protsessi ja teha tehaseseadete taastamise. Seejärel on vaja taaskord oodata, et ruuter töörežiimi saavutaks. Tegemist on väga ajakuluka testisammuga, kuid siiski vajalik testprotsesside käivitamiseks.

Enne testiprotseduuride alustamist on testisamm FirmwareVersion. Selle sammu eesmärgiks on kontrollida sobiva tarkvara olemasolu ning vajadusel selle uuendamine. Sellel meetodil on viis testplaani parameetrit ja kaks plaanimuutujat. Plaanimuutuja on üldjuhul mõeldud arendajale. Kõiki parameetreid ei soovita näidata testri kasutajale. Oodatav parameeter on fail, mis laetakse vajadusel peale. Lisaks viiteaeg ja ajalimiit selle protsessi läbiviimiseks ning kas üldse soovitakse uuendust läbi viia. Plaanimuutujaks on süsteemi serveris asuva faili asukoht. Ühes keskkonnas võib olla palju testreid mida haldavad serverid ning seetõttu hoiustatakse tarkvara uuendusfaile seal. Viimaseks plaanimuutujaks on SeleniumResultIndex, mis määrab ära Python rakenduseset tagasi tulevate väärtuste loogika. Meetodi alguses veendutakse, et seadmel on olemas LAN liideses IP. Selle alusel saame järeldada, et ühendus on endiselt olemas. Järgnevalt saadetakse käsk tarkvara versiooni pärimiseks. Kui tarkvaraversioon on sobiv või uuendust ei oodata, saame siinkohal testisammu lõppenuks lugeda. Juhul kui on vaja rakendada uuendust tuleb kasutada RemoteFileDownloader teeki, et tarkvara testrisse installida. Ebaõnnestunud installimise korral tagastatakse veaolukord. Eduka installimise korral saab jätkata tarkvarauuendusega. Selleks saadetakse Selenium serverile Update käsk koos faili nimega. Järgnevalt saab ära kasutada varasemalt loodud funktsioone seadme aktiivsuse kontrollimiseks. Kui seade on edukalt töörežiimis tuleb taaskord kontrollida, et uuendus on rakendunud.

Testi lõppedes tuleb taaskord rakendada tehase andmetele lähtestamist. Seda saab samuti rakendada eelnevalt loodud funktsioonide abil. Tehase andmete lähtestamist saab kontrollida SSID abil. See väärtus peab olema mõlemal sagedusel samaväärne. Testi viisakaks lõpetamiseks rakendatakse StopWebService, mis prindib logisse Selenium rakenduses toimunud käskude järjekorra. Lisaks saadetakse sulgemis signaal.

3.2 Klass Sercom

Klassi eesmärk on lahendada tootega seonduvaid probleeme ning äriloogikast tulenevaid vajadusi. Klass pärineb MainTestPlan, General ja WebTests baasklassidest. Klassi pärinemine on tingitud rakenduse arhitektuurist. MainTestPlan klass on baasklass iga testiplaani klassi loomisel ehk iga toote grupiga seonduva klassi baasklassiks. MainTestPlan pärineb virtuaalselt TestPlanParenti klassist. General sisaldab üldisi meetodeid ning muutujaid, mis on laialdaselt levinud kasutusalaga. Näiteks tootele vooluallika lülitus mehhanism. Viimaseks pärilikkuseks on WebTests, mis sai loodud toetama antud toote lahendust. Tegemist on kõrgema klassiga, sest see vastutab kogu suhtluskanali eest.

Klassi konstruktor sisaldab defineeritud kujutist tulemustest. Tegemist on sõne ja lambda tüüpi muutujaga *m_functions,* mille alusel moodustatakse tulemused (Joonis 5).

```
QMap<QString, void (Sercom::*)(const FunctionResult&, const
QList<TestParameterItem>&)>
 m_functions;
```

Joonis 5. Tulemusmuutuja

Enne testide alustamist on mõistlik veenduda, et Mikrotik oleks õiges seadistuses. Alguses käivitatakse funktsionaalsus SetupMikrotik, mis seadistab LAN pordid aktiivseks. See tagab kindluse, et test käivitatakse alati samas olekus. Juhuks kui keegi on seadistust muutnud või mingil põhjusel on jäänud eelmisest testi tsüklist sobimatu olek. Tegemist on lihtsa konfigureerimisega ning LAN1 liidese aktiveerimiseks rakendub vastav kood (Joonis 6).

```
m_mikrotik.setInterfaceState(Mikrotik::InterFace::LAN1, true);
```

Joonis 6. Esimese liidese aktiveerimine

Seadme üldise info kogumiseks ja tulemuse väljastamiseks on autor loonud testi osa GeneralInfo. See on informatiivne testisamm, mis kontrollib skaneeritud väärtust. Selle abil saab veenduda, et toote peal olev silt on korrektne. Võrdleb seadme poolt väljastatavat seerial numbrit ja operaatori poolt skaneeritud väärtusi. Info saamiseks peab antud toote puhul pöörduma Selenium serveri muutuja poole. See muutuja saadab käsu Python skripti, mis tagastab fikseeritud formaadis vastuse. Kui tagastatud vastus ei ole sobilik, siis korratakse tegevust. Selleks kasutatakse tsüklit, et vältida koodikorduvust. Vajadusel teeb kuni kolm kordust. Põhjuseks on tihtipeale toodete

ebastabiilne käitumine ning ettevõtte siseselt on vastu võetud otsus, et suhtlust korratakse kuni kolm korda.

Järgides Robert C Martini soovitusi raamatust “Clean Code”. Koodi kordumist ei tohiks esineda[10].

Järgnevaks sammuks saab olema WifiProvisioning ehk WiFi konfigureerimine. Võimalikult varakult konfigureerimine annab seadmele aega rakendada seadistused. Toote seadistamiseks on taaskord vajalik kasutada Seleniumit. Eesmärk on tootest välja lugeda WiFi parool, et saaks seda tulevikus WiFi testis kasutada. Samuti on vajalik SSID muutmine. Tihtipeale on tootel tehaseseadete järgi ühine võrgunimetus nii 2.4 GHz jaoks kui ka 5GHz. See on tingitud sellest, et kodukasutajatel oleks mugavam. Koduseade saab kasutada mõlemat võrku ilma, et peaks vahepeal võrguühendust kaotama. Seade ise otsustab, kumb talle paremini sobib. Seda võimekust nimetatakse sageduse suunamiseks (*Band Steering)* ning on IEEE standard. Tester peab veenduma mõlema liidese töös ning seetõttu on SSID muudetud. See annab läbipaistvust, mis liidest parasjagu kontrollitakse.

WanPort testi samm. WAN liidese testimine toimub Mikrotiki abil. Mikrotikist avatakse laivõrgu liidese ühendus ning veendutakse, et seal on 1 Gbps ühendus. Seda saab kontrollida Mikrotiki võimekusega.

InteractiveTest samm tegemist on operaatori testi sammuga. Antud testri olekul puudub riistvaraline võimekus täisautomaatseks testiks ning seetõttu on vajalik operaator, kes valideeriks LED ja nuppude töökorda. Testi samm on ise küllaltki lihtsakoeline. Küsitakse operaatorilt seadme seisukorra kohta. Nupuks on WPS nupp, mis töökorra puhul käivitab LED tule. Selle abil saab mõlemad olekud ära kontrollida. Operaatori sõnumi saatmiseks on kasutusel Localizator teek, mis võimaldab sõnumit saata kohalikus keeles. Tõlkega sõnumid on salvestatud JSON failis ning Localizator teek ülesanne on korrektne sõnum JSON failist üles leida. See testsamm jääb vahele juhul, kui tester on tsükkelrežiimis. Sellisel juhul pannakse tulemus õnnestunuks ja kommentaar tulemusele, et see on vahele jäänud.

WPS-i töö eesmärk on võimaldada lihtsamalt seadet ühendada WiFi võrku. Üldjuhul kasutatakse printerite ja teiste IoT ühenduste loomise jaoks.

3.3 Üldised testsammud

Need sammud ei ole autori poolt loodud, aga vajadusel on autor täiustanud olemasolevat funktsionaalsust. Tegemist on levinud testisammudega, mis on paljude seadmete valideerimiseks kasutusel ning koondatud ühisesse klassi General. Selle projekti raames kasutatakse General klassist järgnevaid meetode: InitTest, GetValuesFromCsv, CloseDrawer, SshMikrotik, SetRouting, PowerUp, USBVoltage, EthernetLinkTest, WiFiTwoPointFour, WiFiFive, PowerDown.

InitTest laseb testrioperaatoril skaneerida seadme pealt triipkoodi, et saaks hiljem valideerida selle korrektsust. Seadme seerianumber on oluline identifitseerimiseks ning selle alusel saab seadme kohta lisainfot pärida. Skaneeritud väärtuse õigsust kontrollitakse regulaaravaldise mustri alusel[11]. Meetod saab mustri parameetrina, mis on testplaani JSON failis. Regulaaravaldise muster on autori poolt koostatud ning seerianumbri muster on 2[0-9]{3}DFR[0-9]{6} ja MAC aadressi muster on [A-F0-9]{12}. Seerianumbri mustri loomisel tugines autor arenduseks antud toodete seerianumbritest. Kolme toote sarnasusteks oli algav number 2 ja järgmised kolm numbrit, millele järgnes DFR ning seejärel kuus numbrit. MAC aadressi muster kontrollib, et tegemist oleks MAC aadressile omaste sümbolitega ehk A kuni F täheni või numbrid nullist üheksani. Juhul kui skaneeritud väärtus ei vasta reeglitele palutakse operaatoril uuesti proovida.

GetValuesFromCsv samm teeb ühenduse testreid haldavasse serverisse, kus paikneb CSV fail. Tegemist on andmebaasi simuleeriva failiga, mis võib sisaldada sadutuhandeid kirjeid. Failist otsitakse infot seerianumbri alusel. Info kiireks leidmiseks kasutatakse operatsioonisüsteemi võimekust ja kasutatakse Grep programmi. Grep programm on väga võimekas ja suudab kiirelt leida failist otsitavat rida. Autor on täiustanud olemasolevat meetodi lisades parameetritele juurde indeksid, mille alusel õigeid väärtusi leida ja genereerida tulemusi. Neid tulemusi kasutatakse hiljem seadme testides, antud projektis sisaldab Grep vastus veebiliidese parooli, WiFi SSID ja WiFi parooli.

CloseDrawer on operaatori testsamm, kus tuleb operaatoril sulgeda testri sahtel. Sellega tagatakse kvaliteetne testkeskkond.

SshMikrotik samm uuendab vajadusel Mikrotik konfiguratsiooni.

SetRouting seadistab autori poolt kirjutatud ruutingud.

PowerUp käivitab toote ja mõõdab seadme vooluallika tarbimist.

USBVoltage samm mõõdab USB pesast voolu. Autor on seadistanud tarbimislimiidi parameetriks 5 volti. USB töökorrasolekuks ei piisa ainult voolumõõtmisest, sest andmeliikumist ei valideerita. Antud toote kasutajaliides, aga ei võimaldanud andmeliiklust teostada.

EthernetLinkTest kontrollib LAN pesade korrasolekut. Porte kontrollitakse ükshaaval, et ei tekiks Mikrotiki poolseid probleeme, kus paketid hakkavad läbi teiste liideste liikuma. Mikrotikist kontrollitakse ühenduse 1Gbps võimekust ning ühenduse olemasolu Ping programmi abil.

WiFiTwoPointFour ja WiFiFive sammud on analoogsed, üks on 2.4GHz võrgu jaoks ja teine 5GHz võrgu testimiseks. Testisammus kasutatakse varasemalt saadud infot ning seadistatakse Mikrotik konfiguratsioonis võrguparool. Korrektselt seadistatud Mikrotik ühendub automaatselt ligipääsupunkti. Võrguühenduse kontrolliks kasutatakse taaskord Ping rakendust ning mõõdetakse RSSI väärtust. Projektis on kiirustestid kliendisoovil välja lülitatud.

PowerDown samm lülitab toote välja, mis tagab ohutuse ja voolutarbimise vähenemist.

4 Python rakendus

Rakendus kasutab Socket teeki, mis on sisse ehitatud programmeerimiskeeles Python. Socket võimaldab luua TCP serveri, mille kaudu saab saata Python rakendusele käske. Selleks, et serverisse oleks võimalik mitmeid käske saata on kasutusel lõppmatustsükkel. Server kuulab ühendusi kuni saab ühendust sulgeva käsu. Python Socket serveri elutsükli eest vastutab C++ rakenduse teek SeleniumServer.[12]

Socket objekti loomisel kasutatakse parameetreid AF_INET ja SOCK_STREAM. AF_INET spetsiifika annab IPv4 tüüpi aadressi ning SOCK_STREAM TCP tüübi. Samuti tuleb lisada lisaväärtused Socket objektile, et aadressi oleks võimalik mitmeid kordi kasutada ja vältida pidevat aadressi kasutus veateadet. Selleks rakendatakse lisaparameetrid socket.SOL_SOCKET ja socket.SO_REUSEADDR[13].

4.1 Selenium

Selenium moodul on kogum tööriistadest, millega on võimalik automatiseerida interneti brauseri käitumist. Sellega saab sooritada kõiki tavapäraseid tegevusi. Otsida mingi elemendi olemasolu, vajutada ning lugeda seda ja samuti sisestada andmeid, olenemata elemendi omapärast. Näiteks kohalikust masinast tarkvara üleslaadimist. Tegemist on ühe liidesega, mis hõlmab populaarsemaid brausereid ning programmeerimiskeeli. Antud projektis on kasutusel Chrome ja programmeerimiskeele Python kombinatsioon.[14]

Selenium käivitab brauseri tasemel draiverid. Autori projektis on tegemist Selenium versioon kolmega ning kasutatakse Chromedriverit. Brauseriks on Chromium, mis on vabavaraline brauseri tarkvara [15].

Brauseri draiveri ja brauseri parameetrite abil saab Selenium WebDriver konstrueerida ning kasutada brauseri võimekust lokaalselt. WebDriver annab võimekuse manipuleerida DOM objekte. Veebielementide leidmisel kasutab autor lokalisaatoriks

xPath lahendust. XPath on olemuselt rohkem masinale loetav, kuid tagab parema efektiivsuse, kui tavaline elemendi id. XPath on kahte erinevat tüüpi, absoluutsed ja relatiivsed viidad. Loogika on sarnane operatsioonisüsteemidega. Absoluutne xPath algab juurikast ning selle nõrkuseks on see, kui vahepealt midagi muudetakse, siis see enam ei tööta. Relatiivne xPath algab HTML DOM keskelt ning seda eristab viite alguses kahekordne kaldkriips (//).[16]

Selenium meetodeid kasutades tuleb silmas pidada, et iga ebaõnnestunud olukord tagastab erindi. Seetõttu peab neid meetodeid alati kasutama erindi tegelemise metoodikaga [17].

4.2 Serverrakendus

Python failis imporditakse Datetime, Signal, Socket, Time ja Seleniumi teeke. Datetime teeki vajab rakendus selleks, et veaolukorras oleks võimalik jäädvustavale faili nimele lisada veahetke aeg. Signal teek initsialiseeritakse selleks, et serverrakendus saaks sulgemis signaale rakendada. Süsteemi poolt saadetud SIGINT, SIGTERM või SIGHUP. Nende signaalide peale rakendatakse sulgemis meetod. Socket teek vastutab ühenduste loomise ning server staatuse loomise eest. Time teek on kasutusel, et saaks rakenduses viiteid kasutada. Selenium teek on suur ning seetõttu ei soovi sealt kõike importida. Selenium teegist piisab webdriver. Selenium.common.exception teegi seest saame veahaldus muutujad. Nendeks on NoSuchElementException, TimeoutException ja WebDriverException. Selenium.webdriver.common moodulist imporditakse By ja Keys. Selenium.webdriver.support imporditakse expected_conditions nimega EC.Selenium.webdriver.support.ui moodulist WebDriverWait. Webdriver on kõige tähtsam, selle abil saab luua brauseri sessiooni. By muutuja abil saab selektoreid valida. Autor on kasutanud xPath selektoreid elementide leidmisel. Keys muutuja abil saab brauserisiseselt klaviatuuri simuleerida. WebDriverWait on kasutusel selleks, et oodata elementide välja laadimine.

Socket peab siduma serveri IP aadressi ja pordiga. Antud juhul on IP *localhost* ehk 127.0.0.1. Port on autori poolt määratud 21002. Portidel on kindel tava ning oodatavad väärtused. Port 21000 on TCP serverile sobiv [18]. Serveri loomise ajalimiidiks seadistatakse 1 sekund. Selle aja möödudes pole server port 21002 seadistatud tuleb veaolukord. Selline veaolukord võib tekkida, kui mõni teine rakendus juba kasutab

sellist porti. Kuulamis perioodiks määratakse 5 sekundit. Järgnevalt võib rakendus luua tester objekti käskude lahendamiseks. Autor soovib server rakendust, mis on võimeline eluea jooksul mitmeid päringuid vastu võtma, seetõttu kasutatakse lõppmatustsüklit ja hakatakse klientühendusi ootama. Selles tsüklis võib tekkida veaolukordi seega on vajalik kasutada erindi tegelemise loogikat *try* ja *except*. Klientühenduse kuulatavaks andmemahuks on seadistatud 1024 baiti. See on piisav kogus rakendusel käskude vastu võtmiseks. Serverisse saabunud baidid konverteeritakse sõne kujul muutujaks ning poolitatakse semikooloni abil. Selline lahenduskäik annab võimekuse saata mitut argumenti korraga. Mitu argumenti on käskudel Pwd, mis seadistab parooli ja Upgrade. Upgrade käsul on vaja argumenti, sest Selenium vajab informatsiooni, mida faili tarkvara uuenduses kasutada. Iga käsk saadab kliendile tagasi vastuse. Vastus koosneb samuti kahest osast. Veakood ja semikooloniga eralduv informatsioon. Käskude jada on kaardistatud ning igale käsule vastab tester objektist vastav meetod. Juhul kui klientrakendus saadab käsu, mida listis ei esinenud, siis trükitakse see info lihtsalt serveri logisse. Server kuulab kliendi poolt ainult korra, mis tähendab, et peale käsu rakendumist sulgub kliendi jaoks ühendus.

Veaolukordades on kaks eri juhtu. Socket.timeout olukorral jätkab server tööd, aga kui tuleb mistahes teine viga, siis see logitakse serveris.

Tester klassi eesmärk on hallata kogu Seleniumi käitumist. Klass omab muutujad. *Debug* muutuja, mis tavaolekus on *False* võimaldab kuvada brauseri ekraani. *Serial, mac, version, password, wifi_24_ssid, wifi_24_mac, wifi_24_channel, wifi_24_band, wifi_5_ssid, wifi_5_mac, wifi_5_channel, wifi_5_band* ja *wifi_key* on informatiivsed muutujad mida kasutatakse C++ rakendusele info edastamiseks. *Password* muutuja kirjutatakse C++ StartWebService poolt üle. Arenduse ja vigade otsimisel on mugav sinna sisestada algväärtus. Make_screenshot muutuja eesmärk on salvestada ekraanitõmmis juhul kui on ootamatu veaolukord (Joonis 7).

```
self.debug = False
self.serial = ''
 self.mac = ''
self.version = ''
self.make_screenshots = True
self.password = '75848750'
self.wifi_24_ssid = ''
self.wifi_24_mac = ''
self.wifi_24_channel = ''
self.wifi_24_band = ''
self.wifi_5_ssid = ''
self.wifi_5_mac = ''
self.wifi_5_channel = ''
self.wifi_5_band = ''
self.wifi_key = ''
```

Joonis 7. Tester klassimuutujad

Konstruktoris kontrollitakse signaalide olemasolu ning vajadusel suletakse server exit_now meetodi abil. Järgnevalt tehakse muutuja *options* ja väärtustatakse sessiooni seadistused. Määratakse ära, et tegemist on Chrome seadistustega ning brauseri binaarifaili asukoht. Klassi muutuja *debug* olek on väär, siis määratakse *headless* olek, mis tähendab, et brauseri kasutajaliidest ei kuvata. Samuti seadistatakse brauseri akna suurus. Need seadistused on argumendiks Chrome WebDriver tööks. WebDriver tagastab sessiooni ning seejärel saab seadistada lehekülje laadimis ajalimiidiks 10 sekundit. Selline limiit on tingitud seadmete omapärasest käitumisest. Seda ajalimiiti ei tohiks normaalses olekus kunagi vaja minna, aga probleemide korral on vaja veateade genereerida. WebDriverWait meetod annab võimaluse oodata ja otsida elementi. Elemendi leidmise korral on võimalus rakendada tegevusi elemendile. Kasutades meetod *click* saab vajutada sellele elemendile. Kasutades *text* meetodit annab teksti kujul selle elemendi väärtuse. Kui üritatakse rakendada meetodit mis antud elemendil puudub, siis tekib erindi olukord.

Klassi meetod handle_exception saab kaasa argumendi, mis on sõne kujul ning mis on kliendile ehk C++ rakendusele tagastatav. Samuti tehakse meetodis ekraanitõmmis ning salvestatakse testrisse. Sellest meetodist tagastatakse ainult veaolukordi seega tagastuskoodiks on 1. Meetod check_boot_up rakendub kui serverile tuleb käsk CheckBootUp. Eesmärk on seadme tööolekut kontrollida ja navigeeritakse kodulehele. Seadmel on staatiline IP 192.168.1.1 seega saame alati pöörduda seadme poole selle aadressi abil. Kontrollitakse sisselogimislehekülje elementi, kui see eksisteerib, saab seadme lugeda töötavaks. Tagastatakse kood 0, kui ei leitud seda elementi, siis vaadatakse kas on juba element nähtav, mis on ainult sisse loginud kasutajal. Sisse

loginud kasutaja elementi leides tagastatakse kood 2 ja infoks juba sisse loginud. Kõikidel muudel juhtudel on tegemist alles ebamäärases olekus seadmega ning saame tagastada koodi 1 ja info.

Log_in_check meetod käivitub kui saabub käsk LogIn. Esialgu kontrollitakse kas kasutaja on juba sisse loginud. Selle tuvastamise korral saab tagastada koodi 0. Vastasel juhul kontrollitakse kas on sisselogimisekraan. Kirjutatakse kasutaja nime väljale admin ning parooli väljale C++ rakenduse poolt saadetud parool. Kasutatakse klaviatuuri omadust ning vajutatakse *enter* klahvi. Oodatakse kuni kümme sekundit sisse logitud ekraani tunnusteni. Veaolukorra juhul saame taaskord vastata veakood 1 ning õnnestunud korral 0. Selline olukord võib tekkida tihti, sest kodukasutajate juurest ruuterid võivad olla kasutaja poolt määratud parooliga.

FactoryReset käsu peale rakendub start_factory_reset meetod. Selles meetodis kasutatakse kolmest tsüklit. Põhjus tuleneb sellest, et võrguühenduse kaotades logib seade välja ning testi tsükli lõpus tehaseseadete lähtestamise käsu ajal võib olla seade välja loginud. Navigeeritakse lehele 192.168.1.1/ pages.html#/statusandsupport/configuration, kui ei õnnestu tehaseseadete lähtestamise lehekülge avada kontrollitakse väljalogimis-elementi ning logitakse taaskord sisse. Peale tehaseseadete lähtestamise nupule vajutust on ühesekundiline viide, et kinnituselement saaks kuvada enne uut kontrolli. Vajutades ka kinnituselemendile nõustuvalt on tehaseseade lähtestamine initsialiseeritud. Erindite korral pöördutakse erindi haldamismeetodi poole, et salvestada ekraanitõmmis.

GetFwVersion käsu peale rakendub get_general_info meetod. Käsk ootab tarkvara versiooni numbrit, kuid tagastatakse kogu üldine informatsioon. See informatsioon asub samal leheküljel ning kogu info saatmine otseselt ei tekita takistusi. Meetod kutsub välja privaatse meetodi __set_data, mis väärtustab kõik informatiivsed klassimuutujad. Selles meetodis kontrollitakse sarnaselt tehaseseadistuse sammus sisse logimist ning seetõttu on ka esimene lehekülje otsimine tsükliliselt lahendatud. Üldinfo väärtustamist tehakse samuti tsüklis ja tingimuseks on, et kõik väärtused on väärtustatud. Iga tsükli iteratsiooni korral uuendatakse lehekülge. GeneralInfo ja GetSsid käskude vastused tulevad sama põhimõtte alusel. Käsu Serial korral tagastatakse klassimuutuja, sest eelnevad meetodid on selle juba väärtustanud. Erindite korral salvestatakse ekraanitõmmis.

Tarkvara uuenduse saavutamiseks saabub käsk Update. Käivitub meetod start_software_update, millel on argument uuenduseks vajalik fail nimi. Pöördutakse

leheküljele 192.168.1.1/ pages.html#/statusandsupport/firmware_update otsitakse tarkvaraversiooni väärtust, et kontrollida navigatsiooni õnnestumises. Väärtustatakse muutuja *file* elemendiga, mis ootab argumendiks failinime koos asukohaga (Joonis 8).

```
file = self.browser.find_element_by_xpath('//*[@id="file"]')
file.send_keys(file_name)
```

Joonis 8. Tarkvarafaili rakendamine veebielemendis

Kasutajaliideses toimiks sama tegevus nupule vajutades tuleks kohaliku masina failide otsing lahti ning peaks otsima õige faili ülesse. Faili sisestamise järel on vaja saata hiirenupu vajutus elemendile ning tuleb kinnitusküsimus uuenduse alustamiseks.

Erindite korral selles meetodis pöördutakse handle_exception poole ning tehakse ekraanitõmmis.

WifiProvisioning käsu korral rakendub wifi_provisioning meetod,mis kasutab kolmest tsüklit. Selle ruuteri arenduse tarkvara, mis toote tootmisel algselt kasutusel oli ja klientide tarkvara oli erinev ning WiFi konfigureerimine käis teisiti. Sellepärast on autor rakendanud lisa logimist selles meetodis. Käivitatakse seleniumLog fail “a” argumendiga, mis lubab faili infot juurde kirjutada. Faili suuruse pärast ei pea muretsema, sest kirjutatud maht on väga minimaalne ning kõvaketta ruumi probleemi ei teki. Otsitakse lehekülge 192.168.1.1/'/pages.html#/wifi/wifi_settings, mis on uuemates tarkvarades eemaldatud. Tegemist oli lisa parameetrite modifitseerimiseks WiFi seadetes. Kontrollitakse klassimuutujate SSID väärtusi ning vajadusel lisatakse need. WiFi parooli välja lugemiseks on vaja enne vajutada nupule, et parool nähtavale ilmuks. Õnnestunud väärtuste lugemise korral tagastatakse need ning veakoodiks saab olema 0. Meetodi tsükli viimasel katse ebaõnnestumise korral tehakse ekraanitõmmis.

5 Testiplaan

Testplaan on JSON formaadis jada testi sammudest, mis on vaja testi käigus teostada. Samuti hõlmab see testiplaani parameetreid ja limiite testi tulemuste tarbeks. Testplaan fail genereeritakse automaatselt sammudest, mis on valitud testplaani generaatori vaatest. Testiplaani läbimisel genereeritakse tulemuste logifail (Joonis 9).

Testi sammud:

- InitTest - süsteemile vajalik samm, kus operaatorilt küsitakse toote skaneerimist. Selle sammu alusel otsustatakse, mis tootega on tegu ning hiljem on ka võimalik kontrollida seadme peal oleva infosildi õigsust.

- GetValuesFromCSV – testi sooritamiseks vajalik samm. Tester küsib testreid haldava serveri käest seadme kohta lisainformatsiooni seerial numbri alusel. Selles sammus saab tester seadme parooli, millega on võimalik hiljem sisse logida.

- CloseDrawer – General klassi samm, kohustab operaatorit sulgema testrit. Sellega tagatakse kvaliteetne testkeskkond.

- SshMikrotik - süsteemisamm, kus kontrollitakse Mikrotiki konfiguratsiooni õigsust.

- SetRouting – süsteemisamm, seadistab kõik vajalikud ruutingud testi läbi viimiseks.

- SetupMikrotik – tegemist on Sercom klassi sammuga. Seadistab Mikrotiki liideseid, et veenduda õiges seadistuses.

- StartWebService – tegemist on WebTests klassis sammuga. Käivitab Python rakenduse ning saadab tööks vajaliku parooli. Selle alusel kontrollitakse ka rakenduste omavaheline suhtlus.

- PowerUp – General klassi kuuluva sammuga. Üldine meetod, mis lülitab vooluallika töökorda ning ootab limiidis määratud aja. Üldjuhul enamus tooted vajavad tööseisu saavutamiseks mitukümmend sekundit või isegi minuteid.

- BootUpAndReset – WebTests klassi samm. Esimene samm tootega suhtluse loomiseks. Kontrollib parooli korrektsust ning tehakse tehaseseadete lähtestamine. Ebaõnnestunud sisse logimise korral kuvatakse operaatorile ülesanne käsitsi teha tehaseseadete lähtestamine.

- FirmwareVersion – WebTests klassi samm. Kontrollib parameetritest määratud tarkvaraversiooni ning vajadusel uuendab. Uuenduse läbiviimiseks installeeritakse haldusserverist korrektne tarkvara.

- GeneralInfo – Sercom klassi samm üldinfo kuvamiseks.

- WifiProvisioning – Sercom klassi samm WiFi seadistus käsu saatmiseks ja tulemuse koostamiseks. Lisaks väärtustatakse klassimuutujaid, et WiFi testis SSID ja parool oleks korrektne.

- WanPort – Sercom klassi samm. Kontrollib WAN liidese töökorda ning genereerib eraldi tulemuse teistest võrguliidestest.

- USBVoltage - mõõdetakse USB voolutaset. Tulemus sõltub limiitidest, oodatav tulemus on 5 volti.

- EthernetLinkTest – General klassi samm. Kontrollitakse võrguliideste töökorda.

- WiFi – 2.4GHz ja 5GHz traadita võrgu liideste kontroll käib sarnaselt. Kasutatakse General klassi sammu

- Interatcive – operaatoritest, tulemus sõltub operaatori tegevusest. Peab valideerima nupu ning LED töökorda

- FactoryReset – WebTersts klassi samm. Testi lõpuks seadistatakse toode tehaseseadetele.

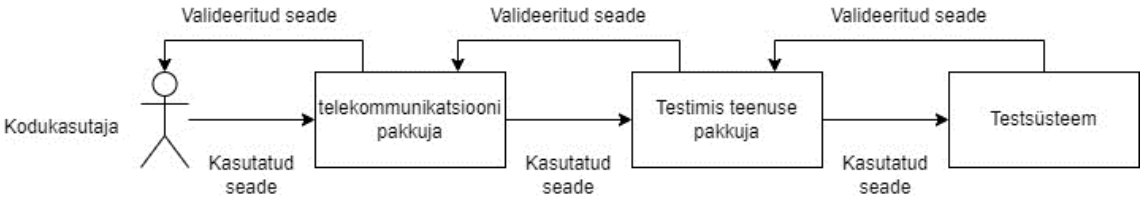
- StopWebService – WebTests klassi samm. Sulgeb Python rakenduse

- PowerDown – General klassi samm, mis lülitab vooluallika välja.

Joonis 9. Tulemuste HTML logi

6 Tulemus ja edasiarendus

Valmis täismahus lahendus seadme testimiseks ning seejärel on loodud võimalus võrguteenust pakkuva seadme uuskasutus (Joonis 10). Uuskasutusega säästetakse ökoloogilist jalajälge ning seadme omanik saab maksimaalse väärtuse tootest. Valideerimistarkvara edasiarendus vajadus sõltub seadme tarkvarauuendustest.



Joonis 10. Tooteahel

6.1 Tulemus

Tulemuseks on poolautomaatne lahendus, mis võimaldab testida seadme erinevaid liideseid. Töö käigus valmis Mikrotik skript võrgu konfiguratsioonide seadistamiseks. Testsüsteemi rakendusele võimekus luua spetsiaalseid testi tulemusi IP3442M seadme jaoks. Sercom ja WebTests testi sammude alusel genereeritakse testi logi HTML ja JSON kujul. Samuti valmis Python rakendus Selenium käskude edastamiseks seadmele. 02.12.2023 seisuga on kliendi poolt üle 26 000 testi sooritatud.

6.2 Edasiarendus

Tegemist on tervikliku lahendusega ning kindlat edasiarendust ei ole planeeritud. Edasiarenduse vajalikkus tekib juhul, kui uuenenud tarkvaraga seadme käitumine muutub. Eelkõige võib muutuda veebilehekülgedel olevad elemendid, mis tähendaks Python skripti uuendamist.

7 Kokkuvõte

Töö annab ülevaate võrguseadme automatiseeritud testimisest ning põhjaliku analüüsi ruuteri liidestest ja funktsionaalsustest. Valmis lahendus Reconext ettevõttele kuuluva testsüsteemi rakendusele lisaväärtus. IP3442M seadme poolautomaat valideerimisvõimekus, mille alusel saab seadet uuskasutusse võtta. Sellega on täidetud algselt püstitatud eesmärk.

Koostati testi läbiviimis kavand tuginedes internetist leitud seadme manuaalile. Kavand on spetsiaalselt disainitud seadme funktsionaalsust arvesse võttes ning selle alusel valideeritakse seadme liidesed.

Eduka testi läbiviimise aluseks on korrektne võrgukonfiguratsioon, mis rakendub Mikrotik seadmele. Mikrotik vastutab IP3442M ja testsüsteemi omavahelise suhtluse eest.

Autor analüüsis erinevaid suhtlusmetoodikaid ning töötas välja suhtluskanali, mille abil on võimalik seadet juhtida. Suhtlustasandiks sai Python rakendus, mis kasutab Selenium teeki.

Täiustati olemasolevat C++ rakendust, et luua võimekus genereerida testi tulemus. Loodi spetsiaalne klass Selenium käskude täitmiseks ning seadmele disainitud klass seadme eripärade lahendamiseks. Lahenduse valmimisel tekkis rakendusele lisaväärtusena klass, mille abil saab tulevikus Selenium suhtlustasandit kiiremini ja mugavamalt kasutusse võtta.

Lahendus on kliendi poolt heaks kiidetud ning kasutusel. Testi on läbi viidud üle mitmekümne tuhande korra.

Kasutatud kirjandus

[1] “Reconext” [Võrgumaterjal]. Saadaval: https://www.reconext.com/about-us/. Kasutatud: 07.11.2023.

[2] “Sercomm IP3442M Quick Start Manual” [Võrgumaterjal]. Saadaval: https://electric.garden/sercomm-p27/ac2600-wi-fi-mesh-router-ip3442m. Kasutatud: 15.11.2023.

[3] “Re-Breaking Wireless Protected Setup” [Võrgumaterjal]. Saadaval: https://www.dais.unive.it/~maccari/files/bibliography/Maccari2013ReBreaking.pdf. Kasutatud: 28.11.2023.

[4] “WPA2 vs. WPA” [Võrgumaterjal]. Saadaval: https://www.lifewire.com/wpa2-vs-wpa-for-wireless-security-3971350. Kasutatud: 30.11.2023.

[5] “Configure IP Addresses and Unique Subnets for New Users” [Võrgumaterjal]. Saadaval: https://www.cisco.com/c/en/us/support/docs/ip/routing-information-protocol-rip/13788-3.html#toc-hId--1212455227. Kasutatud: 01.12.2023.

[6] “Nmap manual” [Võrgumaterjal]. Saadaval: https://linux.die.net/man/1/nmap. Kasutatud: 01.12.2023.

[7] “ACS Automatic Configuration Server” [Võrgumaterjal]. Saadaval: https://intra.optokon.com/intra/product-documents/225/ACS%20Automatic%20Configuration%20Server_2020-07-09.pdf. Kasutatud: 01.12.2023.

[8] “CURL” [Võrgumaterjal]. Saadaval: https://curl.se/. Kasutatud: 01.12.2023.

[9] Martin Fowler, "Patterns of Enterprise Application Architecture".

[10] Robert C. Martin, "Clean Code: A Handbook of Agile Software Craftsmanship".

[11] “Regulaaravaldis” [Võrgumaterjal]. Saadaval: https://pydoc.pages.taltech.ee/extra/regex.html#extra-regex--page-root. Kasutatud 13.12.2023.

[12] “Socket — Low-level networking interface” [Võrgumaterjal]. Saadaval: https://docs.python.org/3/library/socket.html. Kasutatud: 29.11.2023.

[13] “Getsockopt” [Võrgumaterjal]. Saadaval: https://pubs.opengroup.org/onlinepubs/7908799/xns/getsockopt.html. Kasutatud 29.11.2023.

[14] “Selenium documentation” [Võrgumaterjal]. Saadaval: https://www.selenium.dev/documentation/overview/details/. Kasutatud 29.11.2023.

[15] “Chromium browser” [Võrgumaterjal]. Saadaval: https://www.chromium.org/chromium-projects/. Kasutatud: 29.11.2023.

[16] “XPath in Selenium: How to Find & Write?” [Võrgumaterjal]. Saadaval: https://www.guru99.com/xpath-selenium.html. Kasutatud: 29.11.2023

[17] “Erindite töötlemine” [Võrgumaterjal]. Saadaval: https://pydoc.pages.taltech.ee/extra/exceptions/handling-exceptions.html. Kasutatud 29.11.2023.

[18] “Service Name and Transport Protocol Port Number Registry” [Võrgumaterjal]. Saadaval: https://www.iana.org/assignments/service-names-port-numbers/service-names-port-numbers.xhtml?search=21000. Kasutatud: 29.11.2023.

Lisa 1 – Lihtlitsents lõputöö reprodutseerimiseks ja lõputöö üldsusele kättesaadavaks tegemiseks[1]

Mina, Sander Sapp

1. Annan Tallinna Tehnikaülikoolile tasuta loa (lihtlitsentsi) enda loodud teose "Valideerimistarkvara loomine Sercomm IP3442M ruuterile", mille juhendaja on Jaanus Pöial ja kaasjuhendaja Innokenti Sobolev.
   1.1. reprodutseerimiseks lõputöö säilitamise ja elektroonse avaldamise eesmärgil, sh Tallinna Tehnikaülikooli raamatukogu digikogusse lisamise eesmärgil kuni autoriõiguse kehtivuse tähtaja lõppemiseni;
   1.2. üldsusele kättesaadavaks tegemiseks Tallinna Tehnikaülikooli veebikeskkonna kaudu, sealhulgas Tallinna Tehnikaülikooli raamatukogu digikogu kaudu kuni autoriõiguse kehtivuse tähtaja lõppemiseni.
2. Olen teadlik, et käesoleva lihtlitsentsi punktis 1 nimetatud õigused jäävad alles ka autorile.
3. Kinnitan, et lihtlitsentsi andmisega ei rikuta teiste isikute intellektuaalomandi ega isikuandmete kaitse seadusest ning muudest õigusaktidest tulenevaid õigusi.

04.12.2023

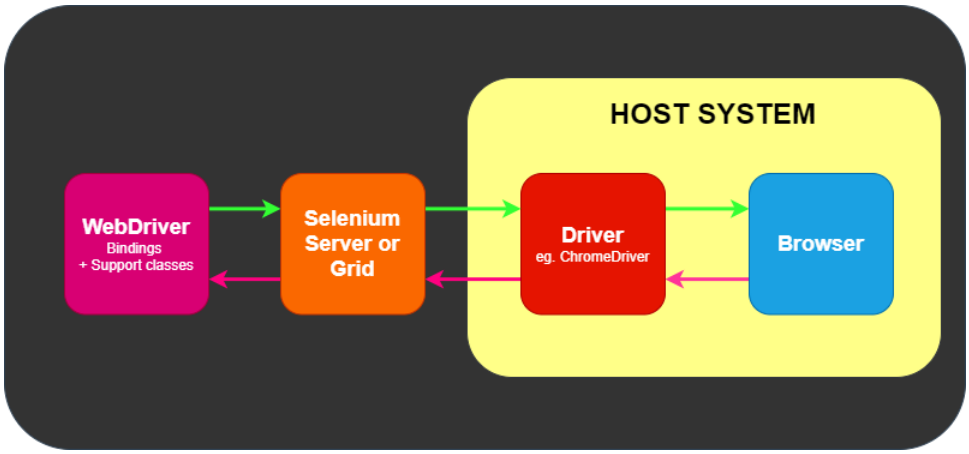
1 Lihtlitsents ei kehti juurdepääsupiirangu kehtivuse ajal vastavalt üliõpilase taotlusele lõputööle juurdepääsupiirangu kehtestamiseks, mis on allkirjastatud teaduskonna dekaani poolt, välja arvatud ülikooli õigus lõputööd reprodutseerida üksnes säilitamise eesmärgil. Kui lõputöö on loonud kaks või enam isikut oma ühise loomingulise tegevusega ning lõputöö kaas- või ühisautor(id) ei ole andnud lõputööd kaitsvale üliõpilasele kindlaksmääratud tähtajaks nõusolekut lõputöö reprodutseerimiseks ja avalikustamiseks vastavalt lihtlitsentsi punktidele 1.1. ja 1.2, siis lihtlitsents nimetatud tähtaja jooksul ei kehti.

Lisa 2 - Tester

Joonis 11. Tester eestvaates

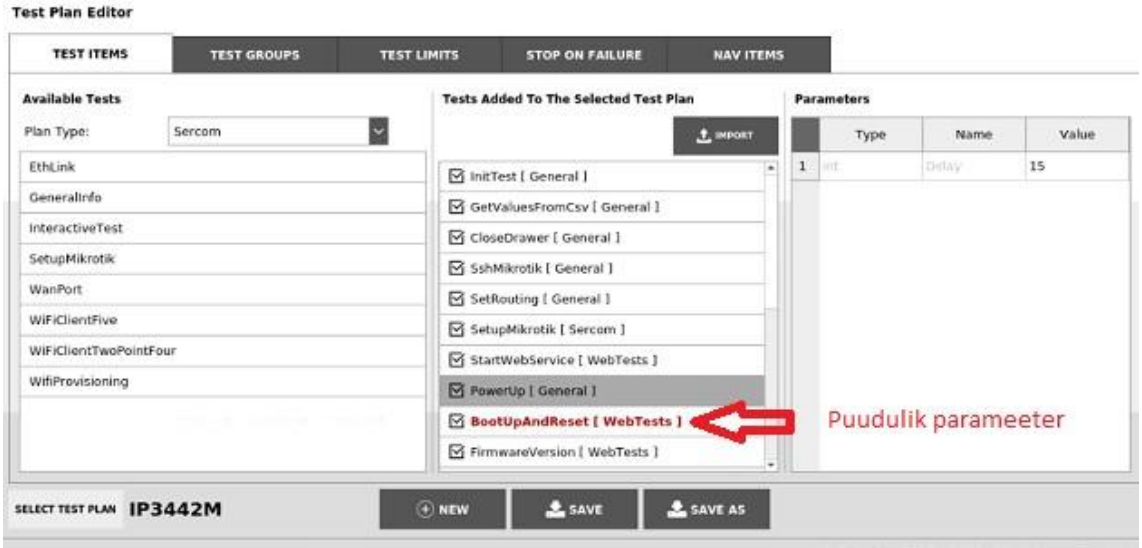
Joonis 12. Tester seest vaates

Lisa 3 - Selenium teegi tööloogika



Joonis 13. Selenium suhtlustasandid

Lisa 4 - Rakenduse testplaani haldus puudulike parameetritega



Joonis 14. Puudulikud parameetrid

Lisa 5 - Rakenduse testplaani haldus

Joonis 15. Parameetrite haldus

Lisa 6 - Mikrotik skript

```
:if ( $RouterModel = "RB4011iGS+5HacQ2HnD" ) do={
       :log info "Applying $RouterModel configuration...";

       :do {

       :log info "Adding wireless security profile...";
       :set ConfStep wlanprof;
              /interface wireless security-profiles
              add authentication-types=wpa-psk,wpa2-psk eap-methods="" group-
ciphers=\
              tkip,aes-ccm management-protection=allowed mode=dynamic-keys
name=\
              profile1 supplicant-identity="" unicast-ciphers=tkip,aes-ccm \
              wpa-pre-shared-key=defaultkey wpa2-pre-shared-key=defaultkey
       :
       :log info "Wireless security profile is added.";

       :log info "Configuring wireless interfaces...";
       :set ConfStep wlanif;
              /interface wireless
              set [ find default-name=wlan2 ] name=wlan2 band=2ghz-b/g/n\
              disabled=no security-profile=profile1 wireless-protocol=any\
              frequency-mode=regulatory-domain antenna-gain=3 channel-
width=20/40mhz-XX
              :delay 100ms

              set [ find default-name=wlan1 ] name=wlan1 band=5ghz-a/n/ac\
              disabled=no security-profile=profile1 wireless-protocol=any\
              frequency-mode=regulatory-domain comment=wlan2 antenna-gain=3
channel-width=20/40/80mhz-XXXX
              :delay 100ms
       :log info "Wireless interfaces are configured.";

       :log info "Configuring WAN interface...";
       :set ConfStep wan;
              /interface vlan
              add name=wanvlan interface=ether6 vlan-id=50
              /ip address
              add address=192.168.34.1/24 interface=wanvlan
              /ip pool
              add name=wanpool ranges=192.168.34.10-192.168.34.100
              /ip dhcp-server
              add name=wanserver interface=wanvlan disabled=no address-
pool=wanpool
              /ip dhcp-server network
              add address=192.168.34.0/24 gateway=192.168.34.1
       :delay 100ms
       :log info "WAN is configured.";
```

```
	:log info "Configuring DHCP clients...";
	:set ConfStep dhcpcli;
		/ip dhcp-client
		add dhcp-options=hostname,clientid disabled=no interface=ether2
		add dhcp-options=hostname,clientid disabled=no interface=ether3
		add dhcp-options=hostname,clientid disabled=no interface=ether4
		add dhcp-options=hostname,clientid disabled=yes interface=ether7
add-default-route=no
		add dhcp-options=hostname,clientid disabled=no interface=wlan1
		add dhcp-options=hostname,clientid disabled=no interface=wlan2
		:delay 100ms
	:log info "DHCP clients are configured.";

	:log info "Configuring NAT...";
	:set ConfStep nat;
		/ip firewall nat
		add action=masquerade chain=srcnat out-interface=ether1
		add action=masquerade chain=srcnat out-interface=ether2
		add action=masquerade chain=srcnat out-interface=ether3
		add action=masquerade chain=srcnat out-interface=ether4
		add action=masquerade chain=srcnat out-interface=wanvlan
		add action=masquerade chain=srcnat out-interface=wlan1
		add action=masquerade chain=srcnat out-interface=wlan2
		:delay 100ms
	:log info "NAT is configured.";

	} on-error={$fnConfError $ConfStep};
	$fnConfDone $ConfName;
	}
```

Joonis 16. Mikrotik skript

Lisa 7 - Klassi WebTests päisefail

```
#ifndef WebTests_H
#define WebTests_H
#include "HeaderFiles/TestPlans/TestPlanParent.h"

class WebTests : virtual public TestPlanParent {
public:
    WebTests();

    void invokeWebFunction(const TestPlanExecutable& tpe, const
FunctionResult& fResult);
    /*
     * {
     * "Name": "WebTests::BootUpAndReset",
     * "Description": "Check if DUT booted up and try to log in. If login
successful perform automatic reset, if not ask for manual reset",
     * "Pass": "DUT booted up. Reset Successful",
     * "Fail": "DUT failed to boot up. Reset failed.",
     * "Error": "Failed to send command to server.",
     * "Parameters": [
     *   "int BootUpTimeout: timeout in seconds for boot up check for single
Ethernet interface",
     *   "int ResetDelay: delay in seconds after factory reset inititalized",
     *   "int EthAmount: amount of eth ports"
     * ]
     * }
     */
    void T_BootUpAndReset(const FunctionResult& fResult, const
QList<TestParameterItem>& parameters);
    /*
     * {
     * "Name": "WebTests::FactoryReset",
     * "Description": "Perform factory reset via selenium server. Check if
WiFi 2.4 GHz SSID changed.",
     * "Pass": "Factory reset performed. WiFi 2.4 GHz SSID changed.",
     * "Fail": "Failed to perform factory reset.",
     * "Error": "Failed to send command to server.",
     * "Parameters": [
     *   "int Delay: delay in seconds after factory reset inititalized",
     *   "int Timeout: timeout in seconds for DUT to boot up after factory
reset",
     *   "int EthEnableDelay: delay in seconds for enabling ethernet port",
     *   "bool LogIn: performs check boot up and log in selenium commands",
     *   "bool PerformVerification: perform SSID check if set to true"
     * ]
     * }
     */
    void T_FactoryReset(const FunctionResult& fResult, const
QList<TestParameterItem>& parameters);
```

```
    /*
     * {
     * "Name": "WebTests::FirmwareVersion",
     * "Description": "Get firmware version. If received FW version does not
match expected, perform FW update and check version again.",
     * "Pass": "FW version matches expected one.",
     * "Fail": "Failed to get FW version. Failed to perform FW update.",
     * "Error": "Failed to send command to server.",
     * "Parameters": [
     *   "string ExpectedFwVersion: expected firmware version",
     *   "string FileName: name of firmware image file",
     *   "int FWdelay: delay in seconds to wait after start of FW update",
     *   "int TimeoutDUT: timeout in seconds for DUT to boot up",
     *   "bool Update: upgrades firmware"
     * ],
     * "PlanVariables": {
     *   "SSFileLocation": "Firmware location in siteserver",
     *   "SeleniumResultIndex": "firmware index of selenium command result"
     * }
     * }
     */
    void T_FirmwareVersion(const FunctionResult& fResult, const
QList<TestParameterItem>& parameters);
    /*
     * {
     * "Name": "WebTests::StartWebService",
     * "Description": "Start selenium script/server. Must be run as
background test. StopWebService must be executed as well in test cycle.",
     * "Pass": "Always.",
     * "Fail": "n/a",
     * "Error": "Failed to send command to server.",
     * "Parameters": [
     *   "string FileName: server script file name",
     *   "int Timeout: Timeout to open the server. If chromium starts slow
errors can occur"
     * ]
     * }
     */
    void T_StartWebService(const FunctionResult& fResult, const
QList<TestParameterItem>& parameters);

    /*
     * {
     * "Name": "WebTests::StopWebService",
     * "Description": "Techical test step for stopping the server started in
StartWebService",
     * "Pass": "Always",
     * "Fail": "n/a",
     * "Error": "Failed to send command to server."
     * }
     */
```

```
    void T_StopWebService(const FunctionResult& fResult, const
QList<TestParameterItem>& parameters);

protected:
    QMap<QString, void (WebTests::*)(const FunctionResult&, const
QList<TestParameterItem>&)> m_webFunctions;
    Selenium m_seleniumServer;
    //factory reset
    int performBootUp(int timeout, int ethAmount);
    void performManualReset(int delay);
    TestResult cmdFactoryReset(int delay);
    TestResult verifyFactoryReset();
    int checkBootUp(int timeout);
    bool doLogin();
    ServerResult performSeleniumCommand(QString command, int timeout = 5, int
retries = 3, int seleniumTimeout = 35);

private:
    TestResult performAutoReset(int delay, int timeout, bool performCheck =
true);
    // Firmware update
    QString getFirmwareVersion(int retries, int fwIndex);
    bool startFwUpdate(const QString& fileName, int delay);
    bool waitForFwUpdateFinished(int timeout);
    bool findLanIp();
};
#endif // WebTests_H
```

Joonis 17. Klassi WebTests päisefail

Lisa 8 - Klassi WebTests kood

https://pastebin.com/tBgWArES

Lisa 9 - Klassi Sercom päisefail

```
#ifndef SERCOM_H
#define SERCOM_H

#include "HeaderFiles/TestPlans/General.h"
#include "HeaderFiles/TestPlans/MainTestPlan.h"
#include "HeaderFiles/TestPlans/WebTests.h"
#include "HeaderFiles/Utils/SeleniumServer.h"

class Sercom : public MainTestPlan, General, WebTests {
public:
    Sercom();

    void invoke(const TestPlanExecutable& tpe, const FunctionResult& fResult)
override;

    /*
     * {
     * "Name": "Sercom::SetupMikrotik",
     * "Description": "Enable eth ports and make dhcp lease for rp362m",
     * "Pass": "always",
     * "Fail": "",
     * "Error": "Missing mikrotik connection or unknown model"
     * }
     */
    void T_SetupMikrotik(const FunctionResult& fResult, const
QList<TestParameterItem>& parameters);
    /*
     * {
     * "Name": "Sercom::GeneralInfo",
     * "Description": "Selenium script data from dut home page",
     * "Pass": "Passes when serial, mac and firmware match",
     * "Fail": "not able to get data or mismatch",
     * "Error": "fails to get data from selenium server. pipe brakes"
     * }
     */
    void T_GeneralInfo(const FunctionResult& fResult, const
QList<TestParameterItem>& parameters);
    /*
     * {
     * "Name": "Sercom::WanPort",
     * "Description": "Wan link test",
     * "Pass": "Link up and 1 Gbps",
     * "Fail": "",
     * "Error": ""
     * }
     */
    void T_WanPort(const FunctionResult& fResult, const
QList<TestParameterItem>& parameters);
```

```
    /*
     * {
     * "Name": "Sercom::WifiProvisioning",
     * "Description": "Gets data from csv or if not possible. Get wifi key
from selenium server(web ui) and change 24 ssid",
     * "Pass": "gets key from ui and changes ssid",
     * "Fail": "",
     * "Error": "does not get response from selenium server"
     * }
     */
    void T_WifiProvisioning(const FunctionResult& fResult, const
QList<TestParameterItem>& parameters);
    /*
     * {
     * "Name": "Sercom::InteractiveTest",
     * "Description": "Gets result from operator task for LED and button
test",
     * "Pass": "operator result",
     * "Fail": "",
     * "Error": ""
     * }
     */
    void T_InteractiveTest(const FunctionResult& fResult, const
QList<TestParameterItem>& parameters);

private:
    TestResult generalInfo();
    QMap<QString, void (Sercom::*)(const FunctionResult&, const
QList<TestParameterItem>&)> m_functions;
};

#endif // SERCOM_H
```

Joonis 18. Klassi Sercom päisefail

Lisa 10 - Klassi Sercom kood

https://pastebin.com/ayucZVMN

Lisa 11 - Python kood

https://pastebin.com/Zpgp2sSt

Lisa 12 - Testplaani JSON


```
{    "NAME": "IP3442M",
    "PLAN_TYPE": "Sercom",
    "TEST_GROUPS": [
    ],
    "TEST_ITEMS": [
        {
            "BLOCK_ON_FAILURE": 3,
            "ENABLED": true,
            "IS_BACKGROUND": false,
            "MY_GROUPS": [
            ],
            "PARAMETERS": [
                {
                    "NAME": "Model",
                    "TYPE": "string",
                    "VALUE": "IP3442M"
                },
                {
                    "NAME": "ReportedModel",
                    "TYPE": "string",
                    "VALUE": "IP3442M"
                },
                {
                    "NAME": "Manufacturer",
                    "TYPE": "string",
                    "VALUE": "Sercom"
                },
                {
                    "NAME": "Barcodes",
                    "TYPE": "string",
                    "VALUE": "Serial_Number,MAC"
                },
                {
                    "NAME": "BarcodeRules",
                    "TYPE": "string",
                    "VALUE": "2[0-9]{3}DFR[0-9]{6},[A-F0-9]{12}"
                }
            ],
            "PLAN_NAME": "General",
            "STOPONFAILURE": true,
            "TEST_NAME": "InitTest"
        },
        {
            "BLOCK_ON_FAILURE": 3,
            "ENABLED": true,
            "IS_BACKGROUND": false,
            "MY_GROUPS": [
```

```
    ],
    "PARAMETERS": [
        {
            "NAME": "FilePath",
            "TYPE": "string",
            "VALUE": "CsvData/SercomRouter/Ip3442M.csv"
        },
        {
            "NAME": "Delimiter",
            "TYPE": "string",
            "VALUE": ","
        },
        {
            "NAME": "Values",
            "TYPE": "string",
            "VALUE": "WiFi SSID,WiFi Key,Password"
        },
        {
            "NAME": "Indexes",
            "TYPE": "string",
            "VALUE": "1,2,0"
        }
    ],
    "PLAN_NAME": "General",
    "STOPONFAILURE": true,
    "TEST_NAME": "GetValuesFromCsv"
},
{
    "BLOCK_ON_FAILURE": 3,
    "ENABLED": true,
    "IS_BACKGROUND": false,
    "MY_GROUPS": [
    ],
    "PARAMETERS": [
    ],
    "PLAN_NAME": "General",
    "STOPONFAILURE": true,
    "TEST_NAME": "CloseDrawer"
},
{
    "BLOCK_ON_FAILURE": 3,
    "ENABLED": true,
    "IS_BACKGROUND": false,
    "MY_GROUPS": [
    ],
    "PARAMETERS": [
    ],
    "PLAN_NAME": "General",
    "STOPONFAILURE": true,
    "TEST_NAME": "SshMikrotik"
},
```

```
{
    "BLOCK_ON_FAILURE": 3,
    "ENABLED": true,
    "IS_BACKGROUND": false,
    "MY_GROUPS": [
    ],
    "PARAMETERS": [
    ],
    "PLAN_NAME": "General",
    "STOPONFAILURE": true,
    "TEST_NAME": "SetRouting"
},
{
    "BLOCK_ON_FAILURE": 3,
    "ENABLED": true,
    "IS_BACKGROUND": false,
    "MY_GROUPS": [
    ],
    "PARAMETERS": [
    ],
    "PLAN_NAME": "Sercom",
    "STOPONFAILURE": true,
    "TEST_NAME": "SetupMikrotik"
},
{
    "BLOCK_ON_FAILURE": 3,
    "ENABLED": true,
    "IS_BACKGROUND": false,
    "MY_GROUPS": [
    ],
    "PARAMETERS": [
        {
            "NAME": "FileName",
            "TYPE": "string",
            "VALUE": "IP3442M.py"
        },
        {
            "NAME": "Timeout",
            "TYPE": "int",
            "VALUE": "60"
        }
    ],
    "PLAN_NAME": "WebTests",
    "STOPONFAILURE": true,
    "TEST_NAME": "StartWebService"
},
{
    "BLOCK_ON_FAILURE": 3,
    "ENABLED": true,
    "IS_BACKGROUND": false,
    "MY_GROUPS": [
```

```
    ],
    "PARAMETERS": [
        {
            "NAME": "Delay",
            "TYPE": "int",
            "VALUE": "15"
        }
    ],
    "PLAN_NAME": "General",
    "STOPONFAILURE": true,
    "TEST_NAME": "PowerUp"
},
{
    "BLOCK_ON_FAILURE": 3,
    "ENABLED": true,
    "IS_BACKGROUND": false,
    "MY_GROUPS": [
    ],
    "PARAMETERS": [
        {
            "NAME": "BootUpTimeout",
            "TYPE": "int",
            "VALUE": "110"
        },
        {
            "NAME": "ResetDelay",
            "TYPE": "int",
            "VALUE": "120"
        },
        {
            "NAME": "EthAmount",
            "TYPE": "int",
            "VALUE": "4"
        }
    ],
    "PLAN_NAME": "WebTests",
    "STOPONFAILURE": true,
    "TEST_NAME": "BootUpAndReset"
},
{
    "BLOCK_ON_FAILURE": 3,
    "ENABLED": true,
    "IS_BACKGROUND": false,
    "MY_GROUPS": [
    ],
    "PARAMETERS": [
        {
            "NAME": "ExpectedFwVersion",
            "TYPE": "string",
            "VALUE": "V2.00.09.002"
        },
```

```
            {
                "NAME": "FileName",
                "TYPE": "string",
                "VALUE": "IP3442M-L-V2.00.09.002-220803.bin"
            },
            {
                "NAME": "FWdelay",
                "TYPE": "int",
                "VALUE": "100"
            },
            {
                "NAME": "TimeoutDUT",
                "TYPE": "int",
                "VALUE": "50"
            },
            {
                "NAME": "Update",
                "TYPE": "bool",
                "VALUE": "true"
            }
        ],
        "PLAN_NAME": "WebTests",
        "STOPONFAILURE": true,
        "TEST_NAME": "FirmwareVersion"
    },
    {
        "BLOCK_ON_FAILURE": 3,
        "ENABLED": true,
        "IS_BACKGROUND": false,
        "MY_GROUPS": [
        ],
        "PARAMETERS": [
        ],
        "PLAN_NAME": "Sercom",
        "STOPONFAILURE": true,
        "TEST_NAME": "GeneralInfo"
    },
    {
        "BLOCK_ON_FAILURE": 3,
        "ENABLED": true,
        "IS_BACKGROUND": false,
        "MY_GROUPS": [
        ],
        "PARAMETERS": [
        ],
        "PLAN_NAME": "Sercom",
        "STOPONFAILURE": true,
        "TEST_NAME": "WifiProvisioning"
    },
    {
        "BLOCK_ON_FAILURE": 3,
```

```
        "ENABLED": true,
        "IS_BACKGROUND": true,
        "MY_GROUPS": [
        ],
        "PARAMETERS": [
            {
                "NAME": "AmountOfPorts",
                "TYPE": "int",
                "VALUE": "1"
            }
        ],
        "PLAN_NAME": "General",
        "STOPONFAILURE": true,
        "TEST_NAME": "USBVoltage"
    },
    {
        "BLOCK_ON_FAILURE": 3,
        "ENABLED": true,
        "IS_BACKGROUND": false,
        "MY_GROUPS": [
        ],
        "PARAMETERS": [
        ],
        "PLAN_NAME": "Sercom",
        "STOPONFAILURE": true,
        "TEST_NAME": "WanPort"
    },
    {
        "BLOCK_ON_FAILURE": 3,
        "ENABLED": true,
        "IS_BACKGROUND": false,
        "MY_GROUPS": [
        ],
        "PARAMETERS": [
            {
                "NAME": "PingAmount",
                "TYPE": "int",
                "VALUE": "4"
            }
        ],
        "PLAN_NAME": "General",
        "STOPONFAILURE": true,
        "TEST_NAME": "EthernetLinkTest"
    },
    {
        "BLOCK_ON_FAILURE": 3,
        "ENABLED": true,
        "IS_BACKGROUND": false,
        "MY_GROUPS": [
        ],
        "PARAMETERS": [
```

```
            {
                "NAME": "PingAmount",
                "TYPE": "int",
                "VALUE": "4"
            },
            {
                "NAME": "PerformThroughputTest",
                "TYPE": "bool",
                "VALUE": "false"
            },
            {
                "NAME": "ThroughputTestDuration",
                "TYPE": "int",
                "VALUE": "30"
            }
        ],
        "PLAN_NAME": "General",
        "STOPONFAILURE": true,
        "TEST_NAME": "WiFiTwoPointFour"
    },
    {
        "BLOCK_ON_FAILURE": 3,
        "ENABLED": true,
        "IS_BACKGROUND": false,
        "MY_GROUPS": [
        ],
        "PARAMETERS": [
            {
                "NAME": "PingAmount",
                "TYPE": "int",
                "VALUE": "4"
            },
            {
                "NAME": "PerformThroughputTest",
                "TYPE": "bool",
                "VALUE": "false"
            },
            {
                "NAME": "ThroughputTestDuration",
                "TYPE": "int",
                "VALUE": "30"
            }
        ],
        "PLAN_NAME": "General",
        "STOPONFAILURE": true,
        "TEST_NAME": "WiFiFive"
    },
    {
        "BLOCK_ON_FAILURE": 3,
        "ENABLED": true,
        "IS_BACKGROUND": false,
```

```
    "MY_GROUPS": [
    ],
    "PARAMETERS": [
    ],
    "PLAN_NAME": "Sercom",
    "STOPONFAILURE": true,
    "TEST_NAME": "InteractiveTest"
},
{
    "BLOCK_ON_FAILURE": 3,
    "ENABLED": true,
    "IS_BACKGROUND": false,
    "MY_GROUPS": [
    ],
    "PARAMETERS": [
        {
            "NAME": "Delay",
            "TYPE": "int",
            "VALUE": "90"
        },
        {
            "NAME": "Timeout",
            "TYPE": "int",
            "VALUE": "120"
        },
        {
            "NAME": "EthEnableDelay",
            "TYPE": "int",
            "VALUE": "20"
        },
        {
            "NAME": "LogIn",
            "TYPE": "bool",
            "VALUE": "false"
        },
        {
            "NAME": "PerformVerification",
            "TYPE": "bool",
            "VALUE": "true"
        }
    ],
    "PLAN_NAME": "WebTests",
    "STOPONFAILURE": true,
    "TEST_NAME": "FactoryReset"
},
{
    "BLOCK_ON_FAILURE": 3,
    "ENABLED": true,
    "IS_BACKGROUND": false,
    "MY_GROUPS": [
    ],
```

```
            "PARAMETERS": [
            ],
            "PLAN_NAME": "WebTests",
            "STOPONFAILURE": true,
            "TEST_NAME": "StopWebService"
        },
        {
            "BLOCK_ON_FAILURE": 3,
            "ENABLED": true,
            "IS_BACKGROUND": false,
            "MY_GROUPS": [
            ],
            "PARAMETERS": [
            ],
            "PLAN_NAME": "General",
            "STOPONFAILURE": true,
            "TEST_NAME": "PowerDown"
        }
    ],
    "TEST_LIMITS": [
        {
            "MAXIMUM": 150,
            "MINIMUM": 30,
            "TESTNAME": "PowerConsumption",
            "UNIT": "mA"
        },
        {
            "MAXIMUM": 0,
            "MINIMUM": -70,
            "TESTNAME": "SignalStrength2.4",
            "UNIT": "dBm"
        },
        {
            "MAXIMUM": 0,
            "MINIMUM": -70,
            "TESTNAME": "SignalStrength5",
            "UNIT": "dBm"
        },
        {
            "MAXIMUM": 10,
            "MINIMUM": 3,
            "TESTNAME": "USB1Voltage",
            "UNIT": "V"
        }
    ],
    "TEST_NAV_ITEMS": {
    }
}
```


Joonis 19. Testisammude alusel automaatselt genereeritud testplaani JSON

Lisa 13 - Tulemuste HTML logifail

https://pastebin.com/NURV4KZq

Lisa 14 - Tulemuste logifail JSON kujul

https://pastebin.com/KpnUksm9