Probleemstelling & Analyse

HBO-ICT: Systems & Middleware Programming

**Student 1**

Naam: Jelle Groot  
Studentnummer: 500902844

Klas: IC102

Datum: **27-Jun-23**

Versie: 3.0

Inhoudsopgave

[1. Probleemstelling 3](#_Toc132794599)

[2. Deelvragen 4](#_Toc132794600)

[3. Analyse 5](#_Toc132794601)

[4. Programma-structuur en testing 6](#_Toc132794602)

# Probleemstelling

De dataset omvat opgevangen netwerkverkeer. Dit is verkeer naar een webserver van een universiteit. Sommige hosts zijn verbonden met het universiteitsnetwerk en zitten daarom in hetzelfde subnet als de webserver. Andere hosts proberen vanaf een andere locatie de website te bezoeken. De meeste hosts communiceren op de een normale manier met de webserver, maar er zijn ook hosts in deze dataset die minder legale doeleinden hebben. Zo proberen een aantal proberen hosts op de webserver in te breken. Ze proberen de webserver te overspoelen met verzoeken of op een andere manier de webserver uit de lucht te krijgen.

Het probleem is dat de dataset uit meer dan 3000 pakketten aan netwerkverkeer bestaat en niet precies bekend is welke hosts de illegale activiteiten uitgevoerd hebben. Ook weten we niet om wat voor activiteiten het gaat en in welke zin deze gevaarlijk waren/zijn.

# Deelvragen

***Welke protocollen worden er gebruikt door de hosts?***

*Door deze vraag te stellen aan de dataset worden de gebruikte protocollen, gekoppeld aan bijbehorend IP-adres, weergegeven. Hiermee kan gekeken worden of er afwijkende protocollen gebruikt worden die niet bij het normale gedrag van de server horen en welk IP-adres deze verbinding heeft gelegd.*

***Wat is de gemiddelde duur van een communicatiesessie?***

*Door de gemiddelde communicatietijd te achterhalen. Als een communicatiesessie veel langer duurt dan de gemiddelde communicatietijd kan dit als een potentiële bedreiging worden gezien. Hierbij kan er sprake zijn van een host die expres de server lang bezet wilt houden, waardoor deze trager wordt. Door deze vraag te stellen aan de dataset kun je vervolgens losse communicatiesessies analyseren en de tijd vergelijken met het gemiddelde die met deze vraag verkregen is.*

***Wordt er communicatie met de server gezocht buiten werktijden?***

*Aangezien de universiteit niet continu geopend is, zijn er bepaalde tijden waarop veel netwerkverkeer niet zo waarschijnlijk is. Met deze vraag kun je analyseren of er communicatie met de server heeft plaatsgevonden, welk IP-adres de connectie heeft gemaakt en hoelaat dit is geweest. Aan de hand van deze informatie kan overwogen worden of de connectie een bedreiging voor het netwerk is of niet.*

# Analyse

***Welke protocollen worden er gebruikt door de hosts?***

*Om de protocollen te achterhalen heb je de volgende data nodig:*

* *Source\_IP (te vinden in de ip\_layer in de dataset)*
* *Frame\_protocol (te vinden in de frame\_layer in de dataset)*

*Met het source\_IP kunnen protocollen gekoppeld worden aan het IP-adres waar ze bij horen. Het* f*rame\_protocol geeft aan welk protocol er gebruikt is in het pakket.*

*Eerst wordt voor elk pakket gekeken welke protocollen er gebruikt worden, waarna deze worden gekoppeld aan het source\_IP van het pakket. Deze combinatie wordt toegevoegd aan een dictionary als deze daar nog niet in staat.*

*Vervolgens wordt in een aparte functie voor elke source\_IP in de dictionary geprint welke protocollen deze heeft gebruikt.*

***Wat is de gemiddelde duur van een communicatiesessie?***

*Om de gemiddelde duur van een communicatiesessie te achterhalen heb je de volgende data nodig:*

* *Source\_IP (te vinden in de ip\_layer in de dataset)*
* *Destination\_IP (te vinden in de ip\_layer in de dataset)*
* *Ack flag (te vinden in de tcp\_layer onder de ‘tcp\_flags\_tree’)*
* *Fin flag (te vinden in de tcp\_layer onder de ‘tcp\_flags\_tree’)*
* *Time\_relative (te vinden in de frame\_layer)*

*Met de source-IP en Destination-IP kan de tijd gekoppeld worden aan de twee IP-adressen die met elkaar communiceren. Met de Fin en de Ack flag kan worden achterhaald wat het laatste pakket van een communicatiesessie is. Als beide waardes op “1” staan, is dit het geval. Tot slot is de time\_relative de totale duur van de communicatiesessie, welke nodig is om uiteindelijk het gemiddelde te bepalen.*

*Eerst wordt gefilterd op het laatste pakket in de sessie door de eis te stellen dat de ack en fin waardes beide 1 zijn. Als dat het geval is wordt de sessietijd opgeslagen aan de hand van het source IP en destination IP. Hiervoor is gekozen, omdat het source IP met meerdere IP adressen contact kan hebben gehad.*

*Vervolgens worden de tijden van de afsluitende pakketten bij elkaar opgeteld en gedeeld door het aantal afsluitende pakketten om de gemiddelde tijd te bepalen. Deze wordt vervolgens weergegeven in een aparte functie “show\_average\_communication\_length”.*

***Wordt er communicatie met de server gezocht buiten werktijden?***

*Om de communicatiesessies buiten de werktijd achterhalen heb je de volgende data nodig:*

* *Source\_IP (te vinden in de ip\_layer in de dataset)*
* *Frame\_time (te vinden in de frame\_layer in de dataset)*

Met het Source\_IP kan de frametime telkens toegevoegd worden aan de lijst van het Source\_IP in de dictionary. Hierdoor kan per IP worden bijgehouden hoeveel communicatiesessie hij buiten werktijd heeft.

De Frame\_time hierbij is de tijd dat het pakket verstuurd is, waaraan gezien kan worden of deze binnen of buiten werktijd valt.

Eerst wordt gekeken of het Source\_IP al tijden in zijn lijst heeft staan, als dat het geval is wordt de tijd toegevoegd aan deze lijst. Zo niet, dan wordt het Source\_IP toegevoegd aan de dictionary en wordt de frametime daarna aan zijn lijst toegevoegd.

Vervolgens worden de tijden van de lijsten vergeleken met de starttijd en stoptijd van de werktijden. Als de tijden erbuiten vallen worden ze inclusief het Source\_IP toegevoegd aan een lijst genaamd “marked\_frames”.

Als het source\_IP geen communicaties buiten werktijden heeft, wordt deze toegevoegd aan een lijst “ips\_to\_delete”. Vervolgens worden alle source\_IP’s in deze lijst verwijderd uit de oorspronkelijke lijst met alle opgeslagen tijden uit de dataset.

Tot slot wordt in een aparte functie de hoeveelheid connecties per source\_ip buiten werktijd weergegeven. Er is ook een “verbose” optie, waarmee een meer gedetailleerde versie wordt weergegeven waarbij alle tijden worden weergegeven.

# Programma-structuur en testing

*In het programma bestaat één class genaamd DosAnalyser.*

*De eerste functie in de class DosAnalyser is “\_\_init\_\_”. Deze bevat de variabelen: “self”, “dataset\_path”. Hier worden alle variabelen opgeslagen die door alle functies in de class aangeroepen kunnen worden. Deze functie heeft geen output.*

*De tweede functie in de class DosAnalyser is “read\_json”. Deze bevat de variabelen: “self”, “jsonfile(string)”. Het doel van de functie is het openen en inladen van de dataset uit de meegegeven jsonfile. De output van deze functie is in de vorm van een lijst.*

*De enige input die deze functie nodig heeft om te kunnen draaien is een meegegeven dataset die wordt meegegeven in de main functie via de command-line-interface.*

*Bij deze functie worden tests gedraaid. Ten eerste wordt gekeken of de meegegeven dataset wel een lijst bevat. Dit is een positieve test. Ten tweede wordt gekeken of het meegegeven bestand wel bestaat, zo niet al er een FileNotFound error gegeven worden. Dit is een negatieve test.*

*De derde functie is “protocol \_by\_ip”. Deze functie gebruikt de variabele: “self”. Het doel van deze functie is om de gebruikte protocollen in de dataset per IP-adres toe te voegen aan een dictionary. Deze functie heeft een dictionary als output.*

*De vierde functie is “show\_protocol\_by\_ip”. Deze functie gebruikt de variabele: “self”. Het doel van deze functie is om de waardes uit de dictionary van “protocol\_by\_ip” weer te geven. Deze functie heeft geen output.*

*De vijfde functie is “average\_communication\_length”. Deze functie gebruikt de variabele: “self”. Het doel van deze functie is om de gemiddelde communicatielengte uit de dataset te berekenen. De output van deze functie is in de vorm van een lijst.*

*De zesde functie is “show\_average\_communication\_length”. Deze functie gebruikt de variabele: “self”. Het doel van deze functie is om de gemiddelde communicatielengte uit de functie “average\_communication\_length” weer te geven. Deze functie heeft geen output.*

*Bij deze functie worden tests gedraaid. Ten eerste wordt gekeken of de output van de gemiddelde communicatielengte gelijk is aan de waarde die deze hoort te hebben. Dit is een positieve test. Ten tweede wordt gekeken of het programma een foutmelding geeft als de meegegeven variabele niet in de vorm van een lijst staat. Dit is een negatieve test.*

*De zevende functie is “worktime”. Deze functie gebruikt de variabele: “self”. Het doel van deze functie is om alle tijden van communicatiesessies buiten werktijden op te slaan in een dictionary gesorteerd op source\_ip. De output van deze functie is in de vorm van een dictionary.*

*De achtste functie is “show\_worktime”. Deze functie gebruikt de variabele: “self”. Het doel van deze functie is om alle communicaties buiten werktijd weer te geven per source\_ip. Dit doet deze functie op basis van de gegevens uit de dictionary van de functie “average\_communication\_length”. In eerste instantie geeft de functie alleen het aantal connecties weer. Er is ook een “verbose” optie, waarbij alle tijden van de communicatiesessies los worden weergegeven. Deze functie heeft geen output.*

*Tot slot is er nog een functie buiten de classes genaamd “main”. Dit is de hoofdfunctie binnen het programma. Deze functie draait door middel van de command-line-interface.*

*De command-line-interface heeft een aantal argumenten. De eerste is een -P argument. Deze hoort bij de functies protocol \_by\_ip en show\_protocol\_by\_ip. Dit argument zorgt ervoor dat de functies protocol \_by\_ip en show\_protocol\_by\_ip draaien.*

*De tweede is een -C argument. Deze hoort bij de functies average\_communication\_length en show\_average\_communication\_length. Dit argument zorgt ervoor dat de functies average\_communication\_length en show\_average\_communication\_length draaien.*

*De derde is een -W argument. Deze hoort bij de functies worktime en show\_worktime. Dit argument zorgt ervoor dat de functies worktime en show\_worktime draaien.*

*De vierde is een -A argument. Deze zorgt ervoor dat alle functies uit het programma gedraaid worden.*

*De vijfde is een -V optional argument. Deze zorgt ervoor dat er indien aanwezig een gedetailleerdere output wordt weergegeven bij het uitvoeren van een van de andere argumenten.*

*De laatste is een positional argument. Deze hoort bij de functie read\_json. Dit argument zorgt ervoor dat er altijd een dataset wordt meegegeven aan het programma.*