Paper: Onderzoeksvoorstel

Scripting voor netwerkbeheer: een geautomatiseerde aanpak voor het beheren van netwerken.

Bachelorproef, 2023-2024

Stijn Coppens

E-mail: stijn.coppens@student.hogent.be
Project repo: https://github.com/stcoppens/Bachelorproef

Samenvatting

Het beheren van netwerken en het reserveren van netwerkadressen gebeurt momenteel grotendeels handmatig, wat inefficiënt is als proces, en tevens gevoelig voor menselijke fouten.
Deze bachelorproef richt zich op het uitwerken van een innovatieve, geautomatiseerde aanpak voor het beheren van netwerken en het toewijzen van netwerkadressen met behulp van scripts. Het hoofddoel van deze bachelorproef is het creëren van een tussenlaag van scripts boven een bestaand beheerprogramma voor netwerken. Als resultaat kan men via een webpagina die beschikbaar is binnen het bedrijfsnetwerk, mits toestemming van de netwerkbeheerder, eenvoudig netwerkadresreservaties aanmaken, wijzigen of verwijderen. De webpagina zal de scripts, die binnen het project geschreven worden, aanroepen om de nodige gegevens op de juiste manier aan te leveren aan het beheerprogramma.
Het verminderen van handmatig beheer van netwerkconfiguraties als resultaat van deze bachelorproef, zal leiden tot efficiëntiewinsten, tijdsbesparingen en een vereenvoudigde aanpak.

Keuzerichting: System & Network Administrator
Sleutelwoorden: DNS, DHCP, IPAM, Python

Inhoudsopgave

1. Inleiding

In de huidige wereld van technologie is het goed beheren van netwerken van groot belang voor bedrijven. Traditioneel gezien kost het handmatig beheren van netwerkinstellingen, zoals het toewijzen van specifieke internetadressen, veel tijd en kan het leiden tot inefficiënties en fouten.

Om dit aan te pakken, zijn er softwareprogramma's die netwerkbeheerders helpen door taken te automatiseren. Dit onderzoek gaat bekijken hoe je met behulp van scripts een soort tussenlaag kunt maken. Hiermee kunnen veelvoorkomende handelingen via een webapplicatie worden doorgegeven aan het systeem dat de netwerkadressen regelt.

1.1. Probleemstelling

Deze bachelorproef zal uitgevoerd worden bij Universiteit Gent (UGent), directie ICT. Momenteel werkt UGent met scripts die op basis van zogenaamde *subnetbestanden* de nodige acties doen om het netwerk te beheren.

Deze subnetbestanden stellen elk een subnetwerk (een aantal opeenvolgende netwerkadressen) voor en beschrijven cruciale informatie zoals belangrijke naamservers, welk *Virtual Local Area Network (VLAN)* nummer, gateway, etc. Hiernaast bevatten deze zowel alle beschikbare als gereserveerde netwerkadressen met daarbij eventueel enkele regels voor domeinnamen en beveiliging.

Voor elke netwerkadresreservatie die moet gebeuren, krijgt het netwerkteam via een webportaal van intern UGent-personeel een mail met daarin de nodige informatie die ze in het daarvoor bestemde subnetbestand plakken. In sommige gevallen dient de netwerkbeheerder eerst zelf nog uit te zoeken welk subnetbestand no-

dig is op basis van de documentatie en opzoekwerk. Indien de aanvrager van de reservatie dit heeft meegegeven in een veld voorzien voor commentaar, moet de netwerkbeheerder de eventuele domeinnaam- of beveiligingsregels zelf nog toevoegen aan de reservatie.

Het overzicht van de beschikbare subnetwerken is beschreven in een interne wikipediapagina met daarbij de beschrijving van elk subnetwerk. Deze huidige aanpak brengt meerdere uitdagingen met zich mee:

- **Tijd**: Het manueel onderhouden van de scripts, subnetbestanden, netwerkadresreservaties (maken en opkuisen) kan veel tijd vragen.
- **Schaalbaarheid**: Doordat elke wijziging het bestaande bestand overschrijft en er dus geen historische data is kan men moeilijk trends herkennen. Ook wikipediapagina's moet men manueel bijwerken bij grote wijzigingen in de structuur.
- **Consistentie**: De huidige aanpak vraagt meerdere manuele acties, wat deze aanpak vatbaar maakt voor menselijke fouten of vergissingen.
- **Beveiliging**: Het bewaren van netwerkadres gegevens in ongecrypteerde bestanden kan leiden tot misbruik.

UGent is momenteel stappen aan het ondernemen voor het implementeren van *Efficiënt IP (EIP)*, een IPAM-softwarepakket, in hun opzet. Dankzij deze implementatie is de verwachting dat de hierboven beschreven indicatoren zullen verbeteren.

1.2. Doelstelling

Deze bachelorproef zal een abstractielaag maken boven EIP waarbij scripts via de *Application Programming Interface (API)* van EIP commando's zullen uitvoeren op EIP. Door de omvang van het EIP-project is het binnen de voorziene tijd van de bachelorproef niet haalbaar om UGent volledig over te zetten op de werking van EIP. Aangezien dit kritische componenten zijn, zal alles eerst uitvoerig getest worden waarbij elke stap in de migratie naar EIP weloverwogen is.

Daarom stel ik als doel om een eerste versie op te leveren van een webportaal waarop men reeds één of meerdere netwerkadresreservaties kan aanmaken, wijzigen of verwijderen. Deze aanvragen komen in een duidelijk overzicht terecht waar de netwerkbeheerders al dan niet openstaande reservaties kunnen wijzigen, goedkeuren of weigeren. Na goedkeuring worden de scripts gebruikt om de reservaties toe te passen in EIP.

Dit project geeft een antwoord op de vraag: Hoe kunnen netwerkbeheerders hun werk vereenvoudigen door het gebruik van scripts? Het beoogde resultaat is:

- Het vereenvoudigen van veelvoorkomende taken, zoals het reserveren van internetadressen.
- Tijd besparen door handmatige handelingen te vermijden.
- Efficiënter werken door menselijke fouten te voorkomen.
- De gebruiksvriendelijkheid verbeteren.

Deze verbeteringen zullen helpen bij het optimaliseren van de netwerkinfrastructuur.

2. Literatuurstudie

Internet Protocol (IP) is het fundament van elk gestructureerd, goed functionerend en veilig netwerk. Het geeft de mogelijkheid efficiënt gegevens te routeren, netwerken te verdelen in meer beheersbare eenheden, toegang te beperken tot gevoelige data of systemen, services te identificeren en het oplossen van netwerkproblemen (Postel, 1981). Dit hoofdstuk legt uit wat *Domain Name System (DNS)* en *Dynamic Host Configuration Protocol (DHCP)* is, waarom IPAM helpt bij het beheren van IP netwerken en waarom HTTP nodig is om te communiceren met EIP.

2.1. DNS

Mockapetris (1987) schrijft dat DNS een systeem is dat *resource records* gebruikt om onder andere vertalingen te voorzien tussen domeinnamen en IP-adressen. Als voorbeeld kan je via de browser naar google surfen via het IP-adres *142.251.36.35* of via het domeinnaam *www.google.be*.

Zoals beschreven door Mockapetris (1987) voorziet DNS meerdere types resource records die netwerkbeheerders kunnen meegeven:

- **A**: Dit resource record beschrijft een host adres. Vb. *"server1.voorbeeld.com. IN A 192.168.1.1"* maakt de vertaling zodat het toestel met de domeinnaam *server1.voorbeeld.com* bereikbaar is zowel via het IP-adres *192.168.1.1* als via de domeinnaam.
- **CNAME**: Dit resource record beschrijft de kanonieke naam van een host, het wordt gebruikt om een alias of subdomein naar het hoofddomein door te verwijzen. Vb. *"www.voorbeeld.com. IN CNAME server1.voorbeeld.com"* zorgt dat server1 ook bereikbaar is via "*www.voorbeeld.com*".

- **MX**: Dit resource record is een *mail exchange* record en wordt gebruikt om aan te geven welke mailservers verantwoordelijk zijn voor het ontvangen van mails binnen een domein. vb. *"voorbeeld.com. IN MX 10 mailserver.voorbeeld.com"* geeft de DNS server mee welke server de mailserver is.
- **NS**: Dit resource record is een *name server* record, het beschrijft welke DNS-servers verantwoordelijk zijn voor het beheren van DNS-informatie voor een domein. Vb. *"voorbeeld.com. IN NS dns1.voorbeeld.com"* verwijst naar *dns1* als DNS-server voor het domein "*voorbeeld.com*".
- **PTR**: Dit resource record is een *Pointer* record, het wordt gebruikt om via IP een vertaling te vragen aan de DNS-server in plaats van via de naam.
- **SOA**: Dit resource record is een *Start of Authority* record die belangrijke informatie bevat over de zone, zoals welke de primaire DNS-server, contactpersonen, etc. zijn.

2.2. DHCP

Dit protocol voorziet een framework voor het doorgeven van configuratie informatie naar hosts (lees: computers) op het netwerk . Zo kan een computer bijvoorbeeld een IP-adres ontvangen waarmee die kan communiceren binnen het netwerk waarop die is aangesloten (Droms, 1997).

IP-netwerken worden door netwerkbeheerders op een logische manier opgesplitst in subnetwerken. Hierbij worden de beschikbare IP-adressen verdeeld in subnetwerken (subnet). Toestellen binnen subnet A zullen elkaar kunnen bereiken terwijl een toestel in een subnet B zonder de nodige routering geen verbinding zal kunnen maken met de toestellen in subnet A.

Voor DHCP zullen netwerkbeheerders subnets (of pools van IP-adressen) aanbieden aan de DHCP-server. Die zal gebruik maken van deze pools door (onder andere) IP-adressen uit te delen aan toestellen die verbinden op het netwerk en daarbij de DHCP-server laten weten dat ze nog geen IP-adres hebben.

Droms (1997) schrijft dat DHCP drie mechanismes gebruikt voor het uitdelen van IP-adressen:

- **Automatisch**: Permanent toewijzen van een IP-adres.
- **Dynamisch**: IP-adres voor een bepaalde tijd toewijzen.
- **Manueel**: Een (door de netwerkbeheerder) vooraf bepaald IP-adres toewijzen, in vakjargon noemt met dit een IP-reservatie.

2.3. IPAM

Naast de vele uitdagingen die zowel DNS als DHCP met zich meebrengen, is het beheren van de vele DNS records, IP-adres ranges en de vaak vele IP-reservaties zeker iets waar een netwerkbeheerder over moet waken. Een mogelijke oplossing hiervoor is het gebruiken van IP Address Management (IPAM) via softwarepakketten die IPAM aanbieden. IPAM laat toe IP-adressen efficiënt te beheren in een netwerk, het leidt tot een gestructureerde aanpak waardoor conflicten tussen subnetten worden vermeden. Het geeft een compleet overzicht van het netwerk met percentages van hoeveel adressen beschikbaar en in gebruik zijn. IPAM geeft eveneens de mogelijkheid om de historiek bij te houden waardoor het van pas komt voor schaalbaarheid en beveiliging van het netwerk (Rooney & Dooley, 2020).

2.4. HTTP

Om communicatie met de API van EIP mogelijk te maken wordt er gebruik gemaakt van *Hypertext Transfer Protocol (HTTP)*. Dit is een client-serverprotocol die communicatie mogelijk maakt op het Internet. Zoals beschreven door Fielding en Reschke (2014) maakt HTTP gebruik van *Uniform Resource Identifiers (URI's)* om unieke webresources te identificeren en bied het verschillende methoden *(GET, POST, PUT, DELETE)* waarmee clients acties kunnen uitvoeren op serverresources. HTTP is *stateless*, elke aanvraag is onafhankelijk, en statuscodes zoals "200 OKën *headers* worden gebruikt om de resultaten en aanvullende informatie van serververzoeken aan te geven, waardoor een gestandaardiseerde communicatie tussen clients en servers mogelijk is.

3. Methodologie

In dit hoofdstuk wordt beschreven welke software het project gebruikt, hoe deze worden toegepast, en welke fases het project zal doorlopen.

3.1. Software

Alle scripts worden geschreven in **Visual studio code** in de programmeertaal **Python**. Voor de testen om rechtstreeks op maat gemaakte commando's te sturen wordt er gebruik gemaakt van **Postman**.

- **Visual Studio Code**: Vanwege de ondersteuning voor meerdere programmeer- en scripttalen, geïntegreerde Git-ondersteuning, debug- en extensie mogelijkheden wordt gekozen om alle scripts in Visual Studio Code te schrijven.
- **Python**: Van Rossum en Drake (2011) beschrijven Python als een eenvoudige, doch krachtige programmeertaal is. De beschikbaarheid van Python-pakketten zoals 'requests'

maakt het mogelijk om efficiënt met gegevens door te sturen naar API's zoals die van EIP. Als interpretatieve taal biedt Python snelle ontwikkeling zonder de noodzaak van compilatie. Python is ook uitbreidbaar, waardoor het eenvoudig is om nieuwe functies toe te voegen. Kortom, Python, met zijn netwerkbibliotheken, vormt een ideale keuze voor dit onderzoeksproject.

- **Postman**: Dit programma geeft de mogelijkheid alle HTTP-verzoeken manueel te maken, versturen en ontvangen. Hierbij is het mogelijk om alle onderdelen van het verzoek te manipuleren en na te gaan wat als resultaat verwacht kan worden bij het uitsturen van bepaalde verzoeken naar de API.

3.2. Methodiek scripts

Voordat een script wordt geschreven, worden eerst gerichte testen gedaan waarbij commando's met parameters naar de API van EIP worden verzonden. Hierbij wordt er zowel naar de resultaten van de API gekeken als naar wat er op EIP zelf gebeurt via de webpagina. Eens deze testen voor een commando afgelopen zijn, wordt er pas overgegaan tot het schrijven van het Pythonscript. Hierbij worden telkens de parameters binnen elk script opgezet volgens de voorwaarden van de API.

3.3. Fase 1. Literatuurstudie

Voor deze eerste fase worden veertien dagen uitgerokken. In deze fase wordt gezocht naar documentatie en literatuur van gelijkaardige projecten waarbij een webpagina met formulieren en logins gebruiken om scripts aan te roepen. Daarnaast wordt er ook uitgebreid aandacht gegeven aan de literatuur van Efficiënt IP zelf om na te gaan welke eisen deze stelt voor het aanmaken, wijzigen en verwijderen van IP-adressen. Op het einde van deze fase zal er een verslag geschreven zijn met alle belangrijke punten die zijn meegenomen uit de literatuur.

3.4. Fase 2. Scripts schrijven

Binnen de tweede fase wordt het grootste stuk van de Python-scripts geschreven. Deze fase voorziet telkens één week om een script te schrijven en daarop aansluitend één week om het geschreven script te testen en te troubleshooten. De drie scripts zijn:

- Maken van een IP reservatie
- Wijzigen van een bestaande IP reservatie
- Verwijderen van een bestaande IP reservatie

3.5. Fase 3. Webpagina schrijven

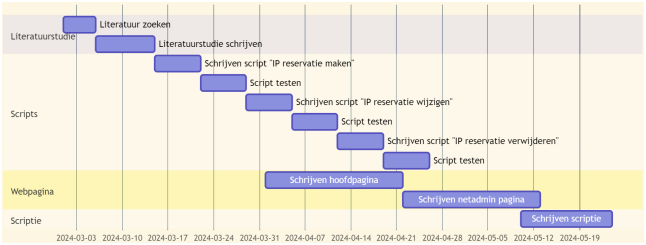
Het schrijven van de webpagina begint halverwege fase twee aangezien deze nauw op elkaar aansluiten. Voor het schrijven van de hoofdpagina van waar de drie scripts kunnen worden aangeroepen worden drie weken voorzien. Hierna zijn er nog drie weken voorzien voor het schrijven van de netwerkbeheerder pagina waarop die de goedkeuring moet geven voor de gevraagde wijzigingen en eventueel zelf nog aanpassingen kan doen.

3.6. Fase 4. Scriptie

De laatste dagen van het project worden gevuld met het bundelen van alle notities uit voorgaande fases en het schrijven van de scriptie.

3.7. Gantt stappenplan

Een overzicht van alle fases worden weergegeven in dit Gantt stappenplan



Figuur 1: Gantt Stappenplan

4. Verwachte resultaten

De verwachte resultaten omvatten een succesvolle eerste implementatie van een webportaal waarop men reeds één of meerdere netwerkadresreservaties kan aanmaken, wijzigen of verwijderen. Deze eerste versie van het intuïtieve webportaal zou een verbeterde gebruikerservaring moeten bieden door middel van geoptimaliseerde API-aanroepen. Verder zou de automatisering van netwerkconfiguraties,
met name IP-adresallocatie, moeten leiden tot verminderde complexiteit, verbeterde efficiëntie en algemene gebruiksvriendelijkheid in het netwerkbeheerproces. Dit zal een belangrijke stap zijn om na het project op voort te bouwen om uiteindelijk een product op te leveren.

5. Discussie, conclusie

Dankzij het implementeren van deze webpagina met de onderliggende kunnen medewerkers van Universiteit Gent eenvoudig, consistent, en snel IP-reservaties maken, wijzigen en verwijderen. De netwerkbeheerders kunnen al deze wijzigingen dan op hun eigen webpagina controleren, aanpassen en beoordelen. Na dit onderzoek zullen er nog voldoende mogelijkheden zijn om de webpagina aan te vullen met extra functies tot de mate dat de netwerkbeheerders zelf geen

wijzigingen meer moeten doen binnen de IPAM tool zelf.

Referenties

Droms, R. (1997). *Dynamic Host Configuration Protocol* (tech. rap.). RFC Editor. https://doi.org/10.17487/RFC2131

Fielding, R. T., & Reschke, J. (2014, juni). Hypertext Transfer Protocol (HTTP/1.1): Semantics and Content. https://doi.org/10.17487/RFC7231

Mockapetris, P. (1987). *Domain names-concepts and facilities* (tech. rap.). https://doi.org/10.17487/RFC1034

Postel, J. (1981). *Internet protocol* (tech. rap.). https://doi.org/10.17487/RFC0791

Rooney, T., & Dooley, M. (2020). *IP Address Management*. John Wiley & Sons. https://doi.org/10.1002/9781119692263

Van Rossum, G., & Drake, F. L. (2011). *An Introduction to Python*. Network Theory Ltd. http://atk.fam.free.fr/fichiers/stage/Python/JF/site/pytut.pdf