Hoe kan trust management effectief worden geïmplementeerd voor het beveiligen van bedrijven met heterogene gesegmenteerde netwerken?

Thibo Haezaert.

Scriptie voorgedragen tot het bekomen van de graad van
Professionele bachelor in de toegepaste informatica

Promotor: Dhr. G. Blondeel
Co-promotor: Dhr. D. Mussen
Academiejaar: 2024–2025
Eerste examenperiode

Departement IT en Digitale Innovatie .

HO
GENT

Woord vooraf

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Samenvatting

Lorem ipsum dolor sit amet, consectetuer adipiscing elit. Ut purus elit, vestibulum ut, placerat ac, adipiscing vitae, felis. Curabitur dictum gravida mauris. Nam arcu libero, nonummy eget, consectetuer id, vulputate a, magna. Donec vehicula augue eu neque. Pellentesque habitant morbi tristique senectus et netus et malesuada fames ac turpis egestas. Mauris ut leo. Cras viverra metus rhoncus sem. Nulla et lectus vestibulum urna fringilla ultrices. Phasellus eu tellus sit amet tortor gravida placerat. Integer sapien est, iaculis in, pretium quis, viverra ac, nunc. Praesent eget sem vel leo ultrices bibendum. Aenean faucibus. Morbi dolor nulla, malesuada eu, pulvinar at, mollis ac, nulla. Curabitur auctor semper nulla. Donec varius orci eget risus. Duis nibh mi, congue eu, accumsan eleifend, sagittis quis, diam. Duis eget orci sit amet orci dignissim rutrum.

Nam dui ligula, fringilla a, euismod sodales, sollicitudin vel, wisi. Morbi auctor lorem non justo. Nam lacus libero, pretium at, lobortis vitae, ultricies et, tellus. Donec aliquet, tortor sed accumsan bibendum, erat ligula aliquet magna, vitae ornare odio metus a mi. Morbi ac orci et nisl hendrerit mollis. Suspendisse ut massa. Cras nec ante. Pellentesque a nulla. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Aliquam tincidunt urna. Nulla ullamcorper vestibulum turpis. Pellentesque cursus luctus mauris.

Nulla malesuada porttitor diam. Donec felis erat, congue non, volutpat at, tincidunt tristique, libero. Vivamus viverra fermentum felis. Donec nonummy pellentesque ante. Phasellus adipiscing semper elit. Proin fermentum massa ac quam. Sed diam turpis, molestie vitae, placerat a, molestie nec, leo. Maecenas lacinia. Nam ipsum ligula, eleifend at, accumsan nec, suscipit a, ipsum. Morbi blandit ligula feugiat magna. Nunc eleifend consequat lorem. Sed lacinia nulla vitae enim. Pellentesque tincidunt purus vel magna. Integer non enim. Praesent euismod nunc eu purus. Donec bibendum quam in tellus. Nullam cursus pulvinar lectus. Donec et mi. Nam vulputate metus eu enim. Vestibulum pellentesque felis eu massa.

Quisque ullamcorper placerat ipsum. Cras nibh. Morbi vel justo vitae lacus tincidunt ultrices. Lorem ipsum dolor sit amet, consectetuer adipiscing elit. In hac habitasse platea dictumst. Integer tempus convallis augue. Etiam facilisis. Nunc elementum fermentum wisi. Aenean placerat. Ut imperdiet, enim sed gravida sollicitudin, felis odio placerat quam, ac pulvinar elit purus eget enim. Nunc vitae tortor. Proin tempus nibh sit amet nisl. Vivamus quis tortor vitae risus porta vehicula.

Inhoudsopgave

Lijst van figuren vi

Lijst van tabellen vii

Lijst van codefragmenten viii

1 Inleiding 1
- 1.1 Probleemstelling . . . 1
- 1.2 Onderzoeksvraag . . . 2
- 1.3 Onderzoeksdoelstelling . . . 3
- 1.4 Opzet van deze bachelorproef . . . 3

2 Stand van zaken 4
- 2.1 Basisbegrippen . . . 4
- 2.2 Verschillende truststores . . . 7
- 2.3 Bestaande tools . . . 11
- 2.4 Infrastructuur studie . . . 13

3 Methodologie 15

4 Proof-of-concept 17
- 4.1 Virtuele omgeving . . . 17
  - 4.1.1 Webserver . . . 18
  - 4.1.2 Windows server . . . 18
  - 4.1.3 CA . . . 19
  - 4.1.4 Vault . . . 19
  - 4.1.5 Employee-client . . . 19
  - 4.1.6 Admin-client . . . 19
- 4.2 Eerste oplossing . . . 19
  - 4.2.1 Oplossing voor Windows end-points door middel van GPO's met root certificaten . . . 19
  - 4.2.2 Oplossing voor Linux end-points met Ansible . . . 20
  - 4.2.3 Pro's en Con's van de eerste oplossing . . . 23
- 4.3 Tweede oplossing . . . 24
  - 4.3.1 Installeren van een Vault server . . . 24
  - 4.3.2 Installeren van een Vault agent . . . 27
  - 4.3.3 Oplossing voor Windows end-points door middel van GPO's met startup scripts en SCCM . . . 28

4.3.4 Oplossing voor Linux end-points met Chef en Vault 30
4.3.5 Pro's en Con's van de tweede oplossing 35
4.4 Overwegingen en aanbevelingen 35
4.4.1 Schaalbaarheid 35

5 Conclusie **36**

A Onderzoeksvoorstel **38**
A.1 Inleiding 38
A.2 Literatuurstudie 39
A.3 Methodologie 41
A.4 Verwacht resultaat, conclusie 42

Bibliografie **44**

Lijst van figuren

2.1 Certificate enrollment. 5
2.2 Chain of trust 6

Lijst van tabellen

Lijst van codefragmenten

1

Inleiding

In moderne bedrijfsomgevingen worden netwerken steeds complexer en diverser. Organisaties bevatten vaak heterogene netwerken, dat wilt zeggen dat binnen het netwerken verschillende soorten apparaten aanwezig zijn die niet allemaal dezelfde besturingssystemen en software gebruiken. Vaak zijn deze netwerken ook opgedeeld in verschillende segmenten om de impact van een eventuele aanval te beperken. Hoewel segmentatie helpt om bedreigingen te beperken, brengt het ook uitdagingen met zich mee op het gebied van toegangsbeheer en welke bronnen men kan vertrouwen en bereiken binnen de verschillende segmenten. Een specifiek probleem binnen deze context is dat niet elk netwerksegment dezelfde root certificates wil vertrouwen, wat een aanzienlijk beveiligingsrisico met zich meebrengt.
Trust management biedt een mogelijke oplossing door dynamische en contextbewuste toegang te faciliteren, waarbij beslissingen worden genomen op basis van factoren zoals gebruikersidentiteit, gedragsanalyse en beleidsregels. Een belangrijke uitdaging is echter hoe de trust stores van endpoints in een netwerk centraal kunnen worden beheerd om dit probleem effectief aan te pakken. De implementatie van een effectief trust management-systeem in een heterogeen, gesegmenteerd netwerk vereist een grondige analyse van de risico's, beleidsmodellen en technologische vereisten.

1.1. Probleemstelling

Vele bedrijven met grotere diverse netwerken zoals heterogene netwerken en gesegmenteerde netwerken hebben vandaag de dag nogsteeds moeite met het centraal beheren van de (root) certificaten die worden vertrouwd door hun endpoints. Dit komt door de verspreide trust stores die gevormt worden door de verschillende gebruikte applicaties en besturingssystemen binnen het netwerk, daarnaast is het

ook belangrijk om het aantal vertrouwde certificaten te beperken om de veiligheid van het netwerk te garanderen. Vele bedrijven hebben een netwerksegmentatie toegepast afhankelijk van hun bedrijfsrollen en -vereisten, maar dit brengt ook uitdagingen met zich mee op vlak van welke segmenten welke certificates vertrouwen en hoe dit kan worden afgedwongen.

1.2. Onderzoeksvraag

Aan de hand van welke combinatie van tools en configuraties kan een trust management-systeem worden opgezet binnen een bedrijf met een heterogeen gesegmenteerd netwerk waarbij de inhoud van de trust stores van endpoints makkelijk en snel centraal kan worden beheerd?
Deelvragen:

- Wat zijn de belangrijkste concepten en technologieën achter PKI en truststorebeheer?
- Wat zijn de verschillende truststores die worden gebruikt door de meest voorkomende besturingssystemen en applicaties? Hoe kunnen deze worden beheerd?
- Welke tools en technieken worden momenteel gebruikt voor truststorebeheer, en wat zijn hun beperkingen?
- Hoe kan de inhoud van de trust stores van endpoints centraal worden beheerd?
- Hoe kan men de vertrouwde certificaten anders beheren binnen verschillende netwerksegmenten?
- Welk architectuurplan zal worden gebruikt om de proof-of-concept virtuele omgeving op te zetten?
- Hoe worden de gevonden tools en technieken geïmplementeerd in de proof-of-concept virtuele omgeving?
- Wat is de effectiviteit van de opgezette proof-of-concept virtuele omgeving?
- Welke aspecten zijn niet opgenomen in de proof-of-concept virtuele omgeving die in realiteit voor problemen kunnen zorgen?
- Welke aanbevelingen kunnen worden gegeven aan bedrijven die een trust management-systeem willen implementeren?

1.3. Onderzoeksdoelstelling

Het doel binnen dit onderzoek is om een proof-of-concept virtuele omgeving op te zetten die systemen met veel voorkomende besturingssystemen en applicaties bevat, alsook netwerksegmentatie waarbij de trust stores van de endpoints centraal kunnen worden beheerd.

1.4. Opzet van deze bachelorproef

De rest van deze bachelorproef is als volgt opgebouwd:
In Hoofdstuk 2 wordt een overzicht gegeven van de stand van zaken binnen het onderzoeksdomein, op basis van een literatuurstudie.
In Hoofdstuk 3 wordt de methodologie toegelicht en worden de gebruikte onderzoekstechnieken besproken om een antwoord te kunnen formuleren op de onderzoeksvragen.
In Hoofdstuk 5, tenslotte, wordt de conclusie gegeven en een antwoord geformuleerd op de onderzoeksvragen. Daarbij wordt ook een aanzet gegeven voor toekomstig onderzoek binnen dit domein.

2

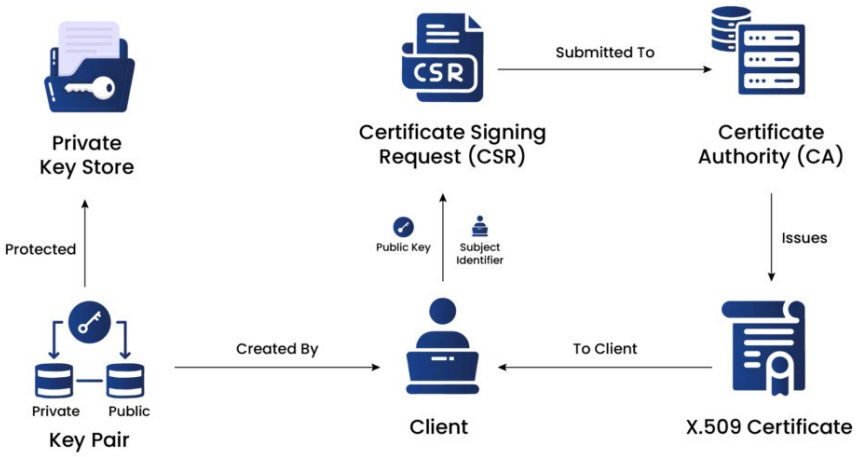
Stand van zaken

2.1. Basisbegrippen

Om het onderzoeksonderwerp samen met de achterliggende uitdaging te begrijpen, is het belangrijk om de werking van een Public Key Infrastructure (PKI) te begrijpen. Thales (2025) definieert een PKI als een set van hardware, software, policies, processen en procedures die noodzakelijk zijn voor het maken, beheren, uitgeven, gebruiken, opslaan en intrekken van digitale certificaten en publieke keys. PKI's zijn de basis die het gebruik van technologieëen zoals digitale handtekeningen en encryptie mogelijk maakt over een grote populatie van gebruikers. Zij helpen namelijk met het vaststellen van de identiteit van personen, apparaten en diensten, wat gecontrolleerde toegang tot systemen en bronnen alsook data beveiliging en controle mogelijk maakt.

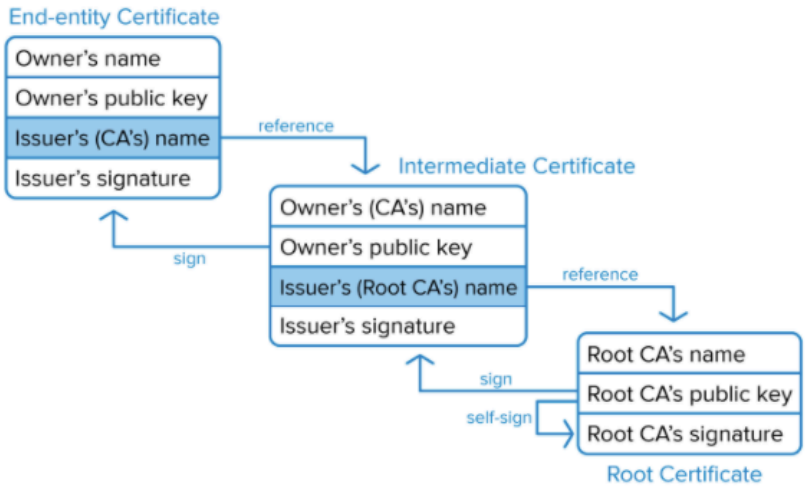
Een groot deel van PKI's zijn Digitale Certificaten en Certificate Authorities (CA's). Een certificate authority is een bedrijf of organisatie die de identiteit van entiteiten (zoals websites, e-mail adressen, bedrijven, individuen, enz.) valideert en ze vastbindt aan cryptografische sleutels door middel van het uitgeven van electronische documenten gekend als digitale certificaten. Een certificaat doet zich voor als een getuigschrift dat de identiteit valideert van de entiteit waaraan het is uitgegeven. (SSL.com, 2024)

Wanneer een certificaat wordt aangevraagd bij een CA, moet de aanvrager eerst een public en private key genereren. De private key moet onder de controle en eigendom van de aanvrager blijven. In sommige gevallen worden de private keys gegenereerd en veilig bewaart in een Hardware Security Module (HSM). (SSL.com, 2024) Om de registratie van certificaten te starten, maakt de aanvrager een Certificate Signing Request (CSR) aan. Deze CSR bevat de public key en andere informa-



Figuur 2.1: Deze afbeelding toont de stappen en componenten in het certificaat uitgave proces.

tie van de aanvrager die in het certificaat zal worden opgenomen, zoals de domeinnaam voor een SSL/TLS certificaat of de aanvrager's e-mail adres voor een S/MIME certificaat. Daarna wordt de CSR ingedient bij de CA. De CA zal de identiteit van de aanvrager samen met de bijkomende informatie verifiëren. De CA kan verschillende manieren gebruiken om de aanvrager zijn identiteit te verifiëren, zoals e-mail verificatie, domein validatie of manuele validatie van juridische documenten. Wanneer de CA het verificatie proces heeft voltooid en vaststelt dat de aanvrager legitiem is, zal het digitale certificaat worden uitgegeven. Het certificaat zal de aanvrager zijn public key en bijkomende informatie bevatten, alsook een geldigheidstermijn en de digitale handtekening van de CA. Het uitgegeven certificaat wordt dan terug gebracht naar de aanvrager, afhankelijk van de CA en het certificaat type zal het certificaat geleverd worden op een verschillende manier zoals via e-mail, een beveiligd portaal of een andere methode. Na het ontvangen van het certificaat is het aan de aanvrager om het certificaat te installeren op de toepasselijke server of apparaat waar het zal worden gebruikt. Als voorbeeld, een SSL/TLS certificaat wordt geïnstalleerd op een webserver voor een beveiligde connectie naar een website. Eenmaal het certificaat is geïnstalleerd, kan het gebruikt worden voor protocollen die verantwoordelijk zijn voor beveiligde communicatie. Clients, gebruikers of andere entiteiten die in contact komen met de certificaat eigenaar kunnen de authenticiteit van het certificaat verifiëren aan de hand van de CA zijn digitale handtekening, wat een beveiligde en betrouwbare connectie verzekerd. Zoals eerder vermeld hebben deze certificaten een geldigheidstermijn (meestal 1 tot 2 jaren). Voor ze vervallen, moet de aanvrager het certificaat vernieuwen via een gelijkaardig proces om het te kunnen blijven gebruiken zonder onderbrekingen. (EncryptionConsulting, 2025)



Figuur 2.2: Deze afbeelding toont de ketting van vertrouwen die de client nakijkt tijdens het verifiëren van een certificaat.

Het verifiëren van een certificaat om de bepalen of het vertrouwd kan worden is belangrijk voor de veiligheid van de communicatie. Okta (2023) zegt dat tijdens een SSL/TLS handshake, de client het certificaat van de server ontvangt. De client controleert of het certificaat nog niet vervallen is en dat de domeinnaam en IP adres op het certificaat gelijkaardig zijn aan dat van de server. Daarna zal de client kijken of het certificaat correct is ondertekend door een vertrouwde CA. In de meeste gevallen zal de server certificate niet ondertekend zijn door de root CA die door de client wordt vertrouwd. In plaats van de root CA zal de client 1 of meerdere intermediate CA's vertrouwen zolang als hun ketting van vertrouwen terug leidt naar een root CA die de client vertrouwd.
Voor elke intermediate CA certificate doet de client hetzelfde verificatie proces waarbij de uitgever (issuer) zijn naam overeenkomt met de certificate eigenaar zijn naam van de volgende certificate in de ketting. Ook wordt de digitale signatuur en public key van het certificaat bekeken om te kijken of deze correct is ondertekend. Dit proces herhaald zichzelf tot de client komt bij een self-signed root CA certificate die de client vertrouwd. Op dit moment heeft de client dan een cryptografische ketting van vertrouwen gemaakt tot de server en kan de SSL/TLS handshake verder gaan.

Bij dit verificatie proces eindigt de client steeds bij een root CA certificaat die door de client moet worden vertrouwd. Om te bepalen welke root CA's door de client worden vertrouwd bestaan er trust stores. Een trust store is een collectie van root certificaten die standaard worden vertrouwd door een client en worden beheerd door bedrijven die de client zijn operating system of browser ontwikkelen, zoals Microsoft, Mozilla en Google. Elke vendor heeft zijn eigen standaarden voor root certificaten maar ze vereisen allemaal dat een uitgevende CA een of meerdere controles

ondergaan om hun vertrouwbaarheid, validiteit en conformiteit vast te stellen via de CA/B Forum Baseline Requirements vooraleer ze worden opgenomen in hun trust store. (Arampatzis, 2020) Over al de trust stores van deze vendors zijn er heel wat certificaten die niet noodzakelijk zijn. Een studie van Perl e.a. (2014) toont dat alleen maar 66% van de certificaten in de trust store van Windows, Linux, MacOS, Firefox, iOS and Android noodzakelijk zijn voor het vertrouwen van websites. Dit zorgt ervoor dat de overige derde van de root certificaten in de trust stores een potentieel veiligheidsrisico vormen voor de client.

Als oplossing hierop kunnen bedrijven het overwegen om deze standaard trust stores af te wijzen. In de plaats daarvan kunnen ze best hun eigen aangepaste, corporate-level trust store maken en gebruik maken van certificate white-listing om te bepalen welke root certificates hierin kunnen opgenomen worden. Dit helpt bedrijven met het aanvals oppervlak te verkleinen door het limiteren van de hoeveelheid vertrouwde CA's en het markeren van niet-vertrouwde SSL/TLS sessies. Organisaties kunnen dan deze certificate whitelist en blacklist updaten op een regelmatige basis afhankelijk van benoodigdheden van hun evoluerende business requirements en groeiend CA landschap. (Arampatzis, 2020)
Het beheren van deze trust stores kan een uitdaging zijn voor bedrijven, zeker bedrijven met heterogene netwerken (netwerken die clients hebben met verschillende operating systemen en browsers). Reddy en Wallace (2010) weerlegt dit door te zeggen dat deze trust anchors (Root CA certificaten) vaak bewaart worden in applicatie-specifieke of OS-specifieke trust stores. Vaak heeft dan 1 machine een verschillend aantal trust stores die niet gesynchroniseerd zijn met elkaar.

De uitdagingen hier zijn dus het vermijden van overbodige vertrouwde root certificaten binnen een netwerk en zijn segmenten afhankelijk van de business requirements en het beheren van deze trust stores over verschillende machines en applicaties.

2.2. Verschillende truststores

Voor het implementeren van een oplossing voor deze uitdagingen is het van belang om te weten hoe de truststores van de verschillende besturingssystemen en browsers werken en hoe deze kunnen worden beheerd.

Windows heeft een truststore die de Trusted Root Certification Authorities store heet. In de officiële documentatie van Microsoft vermeld Microsoft (2024a) dat de Trusted Root Certificate Authorities store op een Windows computer via de Microsoft Management Console (MMC) kan worden beheerd door de Certificate Manager snap-in te gebruiken. De Microsoft Management Console kan worden geopend door het commando `mmc.exe` uit te voeren in deen Run dialog. In het MCC venster

kan men via file -> Add/Remove Snap-in de ”Certificates” snap-in toevoegen. Hierna kan men dan in het MCC venster onder ”Certificates (local computer)” -> ”Trusted Root Certification Authorities” de lijst van vertrouwde root certificaten bekijken en beheren. Microsoft (2024a) vermeld ook dat standaard de Trusted Root Certificate Authorities store geconfigureerd is met een aantal publieke CA’s die de vereisten van de Microsoft Root Certificate Program hebben voltooid.

Naast de GUI manier van beheren van de truststore, biedt Microsoft ook een manier om het te beheren via de command line aan de hand van het `certutil` commando. Certutil.exe is een command-line programma geinstalleerd als onderdeel van de Certitifate Services. Certutil.exe kan gebruikt worden voor het tonen van certificate authority (CA) configuratie informatie, het configureren van Certificate Services en CA componenten te back-uppen en te herstellen. Dit programma kan ook certificaten, key pairs en certificate chains verifiëren. (Microsoft, 2025) Om een certificate store te tonen aan de hand van certutil kan het volgende commando worden gebruikt:

```
certutil [options] -store [CertificateStoreName [CertId
↪ [OutputFile]]]
```

Waar CertificateStoreName de naam is van de certificate store, CertId de match token is van het certificaat en OutputFile de naam is van het bestand waarin de overeenkomende certificaten worden opgeslagen. (Microsoft, 2025)
Om een certificaat toe te voegen aan een certificate store kan het volgende commando worden gebruikt:

```
certutil [options] -addstore CertificateStoreName InFile
```

Waar CertificateStoreName de naam is van de certificate store en InFile de certificate file is die moet worden toegevoegd. (Microsoft, 2025)

Linux heeft niet zoals Windows één centrale trust store. In plaats daarvan heeft Linux verschillende trust stores die afhankelijk zijn van de applicatie of fr distributie van Linux.
Binnen Ubuntu Server moet een certificaat in PEM formaat staan vooraleer het kan worden toegevoegd aan de trust store. Om een PEM-geformatteerd root CA certificaat met als voorbeeld naam ”local-ca.crt” te installeren in de trust store van Ubuntu Server kan het volgende commando worden gebruikt:

```
sudo apt-get install -y ca-certificates
sudo cp local-ca.crt /usr/local/share/ca-certificates
```

```
3  sudo update-ca-certificates
```

Het is hierbij belangrijk dat het certificaat bestand de extensie .crt heeft anders zal het niet worden verwerkt.
De trust store (die gegenereerd wordt door update-ca-certificates) is te vinden op de volgende locaties:

- Als een file (PEM bundel) in /etc/ssl/certs/ca-certificates.crt
- Als een OpenSSL-compatibele certificaat directory in /etc/ssl/certs/ca-certificates.pem

(Canonical, 2025)

Red Hat Enterprise Linux (RHEL) heeft een andere manier van het beheren van de trust store. RHEL biedt de Shared System Certificates aan. De Shared System Certificates storage laat NSS, GnuTLS, OpenSSL en Java toe om een standaard bron te delen voor het ophalen van systeem certificate anchors en black list informatie. Standaard bevat de truststore de Mozilla CA list, die positieve en negatieve trust informatie bevat. Het systeem laat toe om de Mozilla CA lijst aan te passen of om een andere CA lijst te gebruiken. (Red Hat, 2024a)

In Red Hat Enterprise Linux 7 is de systeem-brede truststore te vinden in de directories /etc/pki/ca-trust/ en /usr/share/pki/ca-trust-source/ . De trust settings in /usr/share/pki/ca-trust-source/ worden behandeld met een lagere prioriteit dan de settings in /etc/pki/ca-trust/ . Certificaat bestanden worden anders behandeld afhankelijk van de subdirectory waarin ze worden geplaatst:

- /usr/share/pki/ca-trust-source/anchors/ of /etc/pki/ca-trust/source/anchors/ : voor trust anchors.
- /usr/share/pki/ca-trust-source/blacklist/ of /etc/pki/ca-trust/source/blacklist/ : voor niet vertrouwde certificaten.
- /usr/share/pki/ca-trust-source/ of /etc/pki/ca-trust/source/ : voor certificaten in de extended BEGIN TRUSTED bestandsformaat.

(Red Hat, 2024a)
Om een certificaat in de simpele PEM of DER bestandsformaten aan de lijst van vertrouwde CA's op het systeem toe te voegen, kan je simpelweg het certificaat bestand kopiëren naar de /usr/share/pki/ca-trust-source/anchors/ of /etc/pki/ca-trust/source/anchors/ directory. Om de systeem-brede truststore te updaten ka je het update-ca-trust commando gebruiken zoals volgt:

```
1  cp ~/certificate-trust-examples/Cert-trust-test-ca.pem
   ↪  /usr/share/pki/ca-trust-source/anchors/
```

```
update-ca-trust
```

Om de trust anchors op te lijsten, toe te voegen, veranderen of verwijderen kan het 'trust' commando gebruikt worden. Voor het oplijsten van de trust anchors wordt 'trust list' gebruikt. Een trust anchor opslaan in de trust store kan met het 'trust anchor' sub-commando en het specifiëren van het pad (vb.: 'path.to') naar het certificaat bestand, zoals volgt:

```
trust anchor path.to/certificate.crt
```

Om een trust anchor te verwijderen kan een pad naar het certificaat of ID van het certificaat gebruikt worden:

```
trust anchor --remove path.to/certificate.crt
trust anchor --remove "pkcs11:id=%AA%BB%CC%DD%EE;type=cert"
```

(Red Hat, 2024a)

Applicaties kunnen ook hun eigen trust store hebben. Een voorbeeld hiervan is Mozilla Firefox. Firefox maakt gebruik van de NSS (Network Security Services) library voor het beheren van de trust store. De Network Security Services (NSS) library is een set van libraries ontwikkeld om cross-platform ontwikkeling van veilige client en server applicaties te ondersteunen. De libraries ondersteunen SSL v3, TLS, PKCS 5, PKCS 7, PKCS 11, PKCS 12, S/MIME, X.509 v3 certificaten en andere security standaarden. (Mozilla, z.d.) Firefox geeft bij initiatie een string met pad naar de directory waar NSS de security en configuratie data mag opslaan. NSS slaat 3 bestanden op in die directory:

- cert8.db: slaat publiek toegankelijke objecten op (Certificaten, CRL's, S/MIME records).
- key3.db: slaat private keys op.
- secmod.db: slaat de PKCS11 module configuratie op.

Als in deze directory grote security objecten zitten (zoals grote CRL's), zal NSS deze opslaan in bestanden in subdirectories genaamd 'cert8.dir'. In het geval dat cert8.db en/of key3.db niet bestaan, zal NSS de data lezen van oudere versies vand eze databases (bv.: cert7.db, cert6.db,...) en zal deze data gebruiken om een nieuwe cert8.db en key3.db te maken. (Mozilla, 2018)
Ook beweert Mozilla (2024) dat standaard Firefox op Windows, MacOS en Android zal zoeken en gebruik maken van de third-party CA's die zijn opgenomen in de operating system zijn trust store. Firefox kan geconfigureerd worden om automatisch

te zoeken naar CA's die in de Windows certificate store zijn toegevoegd door een gebruiker of administrator. Dit kan gedaan worden door de `security.enterprise_roots.enabled` optie in

`HKLM/SOFTWARE/Policies/Microsoft/SystemCertificates/Root/Certificates (pad in API flag CERT_SYSTEM_STORE_LOCAL_MACHINE_GROUP_POLICY)`

`HKLM/SOFTWARE/Microsoft/EnterpriseCertificates/Root/Certificates (pad in API flag CERT_SYSTEM_STORE_LOCAL_MACHINE_ENTERPRISE)`

(Mozilla, 2024)

Mozilla (2024) voorziet ook dat enterprise policies kunnen gebruikt worden voor het toevoegen van CA certificaten in Firefox. De 'ImportEnterpriseRoots' key op 'true' zetten, zorgt ervoor dat Firefox root certificaten zal vertrouwen. De 'Install' key zoekt standaard naar certificaten in de onderstaande locaties. Er kan ook een specifiek pad worden opgegeven. Als Firefox daar geen certificaten vindt zal het de standaard directories bekijken:

- Windows
  - %USERPROFILE%\AppData\Local\Mozilla\Certificates
  - %USERPROFILE%\AppData\Roaming\Mozilla\Certificates
- macOS
  - /Library/Application Support/Mozilla/Certificates
  - /Library/Application Support/Mozilla/Certificates
- Linux
  - /usr/lib/mozilla/certificates
  - /usr/lib64/mozilla/certificates

(Mozilla, 2024)

2.3. Bestaande tools

Er bestaan verschillende gratis tools die kunnen helpen met het beheren van truststores over verschillende machines en applicaties. Deze tools zijn niet ontwikkeld specifiek voor het beheren van truststores maar bieden functionaliteiten aan voor het beheren van end-points hun configuratie. Voor Windows bestaat Microsoft System Center Configuration Manager (SCCM).

Microsoft (2024b) zegt dat Configuration Manager deel uitmaakt van de Microsoft Intune familie van producten. De Microsoft Intune familie van producten is een geïntegreerde oplossing voor het beheren van al jouw apparaten. Configuration Manager vebreed en werkt samen met vele Microsoft technologieëen en oplossing zoals onder andere Certificate Services.

Een meer algemene oplossing die werkt voor beide Windows en Linux is Ansible. Ansible is een open-source IT automation engine die provisioning, configuratie management, applicatie deployment, orkestratie en vele andere IT taken automatiseert. Ansible kan gratis worden gebruikt en het project profiteert van de ervanging en intelligentie van zijn duizende gebruikers. (Red Hat, 2025b)
Ansible zijn grootste sterkte is eenvoud. Het heeft ook een sterke focus op beveiliging en betrouwbaarheid, met zo weinig mogelijk bewegende delen. Het gebruikt OpenSSH voor transport (met andere transport en pull modes beschikbaar als alternatieven), en gebruikt leesbare taal die ontworpen is om snel aan de start te kunnen gaan zonder enige training. Red Hat Ansible Automation Platform is een abbonement gebaseerde oplossing die bouwt op de basis van Ansible met een aantal ondernemingsgerichte functionaliteiten. Beide community Ansible en Ansible Automation Platform zijn gebouwt op het concept van een control node en een managed node. Ansible wordt gebruikt op de control node waar bijvoorbeeld een gebruiker een ansible-playbook command uitvoert. Managed nodes zijn de apparaten die worden geautomatiseerd zoals bijvoorbeeld een Windows server. Voor Linux en Windows te automatiseren, connecteert Ansible met de managed nodes en verstuurt het kleine programma's genaamd Andisble modules uit. Deze programma's zijn geschreven als bronmodellen van de gewenste staat van het systeem. Ansible voert dan deze modules uit (standaard over SSH), en verwijdert ze na ze voltooien. Deze modules zijn ontworpen om idempotent te zijn waar mogelijk, zodanig ze alleen een aanpassing uitvoeren op een systeem als dat nodig is. (Red Hat, 2025a)

Een andere tool die mogelijks kan helpen is Open Source Puppet. Puppet is een open source configuratie management tool geproduceert door Puppet Labs. Er kunnen systeem configuraties gedefinieerd worden aan de hand van Puppet's declaratieve taal en deze configuraties kunnen dan opgeslagen worden in bestanden genaamd Puppet Manifests. Puppet kan standalone op een systeem draaien of in een agent/master configuratie. Met standalone Puppet draaien systemen het programma lokaal om hun eigen configuraties aan te passen. Met agent/master Puppet beheert een centrale Puppet master server of servers meerdere systemen die de Puppet agent draaien. Bij beide implementaties worden Puppet manifests gebruikt om systeem configuraties naar de verwachte staat te brengen. Daarnaast bestaat er ook nog een commerciële versie van Puppet genaamd Puppet Enterprise. (Red Hat, 2024b)

Naast Ansible en Puppet bestaan er ook nog andere tools zoals Saltstack en Chef. Gebouwt op Python, is Salt een event-driven automatiseringstool en framework dat is ontworpen om complexe IT systemen te deployeren, configureren en beheren. Salt kan gebruikt worden voor veel voorkomende administratie taken te automatiseren en te verzekeren dat alle componenten van je infrastructuur operatio-

neel zijn in de verwachte staat. Salt heeft vele mogelijke gebruiksdoeleinden, zoals configuratie beheer, wat als volgt inhoud:

- Het beheren van het deployeren van besturingssystemen en hun configuratie.
- Het installeren en configureren van software applicaties en diensten.
- Het beheren servers, virtuele machines, containers, databases, web servers, netwerk apparaten en meer.
- Verzekeren dat de configuratie consistent is en het vermijden van drift.

Salt maakt het mogelijk om applicaties te implementeren en beheren die gebruik maken van elke technologiestack die op bijna elk besturingssysteem draait, inclusief verschilllende soorten netwerkapparaten zoals switches en routers van verschillende leveranciers. Daarnaast kan het ook gebruikt worden voor het automatiseren en orkestreren van routine IT processen zoals veel voorkomende noodzakelijke taken voor geplande server downtime of het upgraden van besturingssystemen en applicaties. Ook kan Salt gebruikt worden om een zelfbewuste, zelf herstellende systemen te maken die automatisch kunnen reageren op uitvallingen, veel voorkomende administratie problemen of andere belangrijke gebeurtenissen. (VMware, 2025)

Progress Software Corporation (2024) noemt Chef een automation company. Vandaag biedt Chef automatiseringsoplossingen voor beide infrasctructuur en applicaties van ontwikkeling tot productie. Chef Infra automatiseert hoe infrastructuur is geconfigureerd, gedeployed en beheerd doorheen het netwerk, ongeacht de grootte.
Chef Workstation maakt het mogelijk om 'cookbooks' te schrijven en je infrastructuur te beheren. Chef Workstation hoort te draaien op de machine die je dagelijks gebruikt, ongeacht of deze Windows, Linux of MacOS draait. Eenmaal het ontwikkelen van je code gedaan is op je workstation, dan kan je deze code uploaden naar de CHef Infra Server. De Chef Infra Server is een centrale opslagplaats voor je configuratie data. Het slaat cookbooks, policies en metadata van elk systeem op. Het communiceren met de Chef Infra Server vanaf een workstation kan via het 'knife' commando. Chef Infra Clients contacteren deze Chef Infra Server periodiek om de laatste cookbooks op te halen. Als de huidige staat van de node niet overeenstemt met wat de cookbook beschrijft, zal de Chef Infra Client de cookbook instrucites uitvoeren. (Progress Software Corporation, 2024)

2.4. Infrastructuur studie

Voor een realistisch beeld te scheppen van een bedrijfsomgeving, zal er gekeken worden naar resultaten van meerdere vragenlijsten en enquêtes die het marktaan-

deel van verschillende technologieëen onderzoeken.

Netcraft (2025) publiceert maandelijks een web server survey waarin ze de marktaandelen van web servers onderzoeken, uit de resultaten van de survey van januari 2025 blijkt Nginx het grootste marktaandeel te hebben binnen alle onderzochte sites met 19,60% alsook alle onderzochte actieve sites met 18,89%. Naast Nginx heeft Apache het tweede grootste marktaandeel met 16,96% van alle onderzochte sites en 17,24% van alle actieve sites.

Binnen de Stack Overflow developer survey van 2024 kan er ook gekeken worden naar veel gebruikte technologieëen binnen de professionele wereld. In de resultaten kan er gekeken worden naar welke van de eerder vermelde tools het meest gebruikt worden in de professionele wereld. (Stack Exchange Inc., 2024) rapporteert onder overige tools dat Ansible gebruikt wordt door 8,1% van de ondervraagde professionele developers, Pupper door 1,1% en Chef door 0,7%.

Binnen de resultaten van de Stack Overflow survey kan er ook gekeken worden naar de meest gebruikte databases. Uit deze resultaten blijkt dat PostgreSQL de meest gebruikte database is met 51,9% van de ondervraagde professionele developers die het gebruiken. MySQL is de tweede meest gebruikte database met 39,4% van de ondervraagde professionele developers die het gebruiken. (Stack Exchange Inc., 2024)

Het bepalen welke DNS server technologie het meest gebruikt wordt binnen bedrijven is moeilijk omdat dit geen publiek beschikbare data is. Hier zal BIND gekozen worden, volgens Internet Systems Consortium, Inc. (2025) is BIND 9 de eerste, oudste en meest gebruikte oplossing waarmee netwerk engineers al meer bekend zijn dan met andere systemen.

E-Soft Inc (2025) publiceert ook rapporten op verschillende technologieëen. Een van deze rapporten bestudeert alle zichtbare mailservers op het internet en kijkt naar welke onderliggende software ze gebruiken. Uit de resultaten van Maart 2025 blijkt dat Exim de meest gebruikte mailserversoftware is met 55,05% en Postfix de tweede meest gebruikte met 38,52%.

Deze informatie zal gebruikt worden voor het bepalen van de gebruikte technologieëen binnen de proof-of-concept virtuele omgeving.

3

Methodologie

Het onderzoek naar centrale truststorebeheer zal opgedeeld worden in drie fases: een literatuurstudie, praktijkstudie en uiteindelijke rapportage en oplevering.
In de eerste anderhalve maand zal de eerste fase van het onderzoek uitgevoerd worden, de literatuurstudie.
Het doel in deze fase is om een diepgaand begrip te krijgen over trust management, certificaatbeheer en de verschillende aanpakken bij truststorebeheer.
In deze literatuurstudie zal er gekeken worden naar de bestaande theorieën, technologieën en andere tools die momenteel worden gebruikt in praktijk.
Ook zal er gekeken worden naar de uitdagingen van trust management in een omgeving met netwerksegmentatie en diverse applicaties en besturingssystemen.
Doorheen deze fase zal er periodiek overleg zijn met de co-promotor om de voortgang te bespreken en een mogelijkse oplossing te kiezen die verder kan worden uitgewerkt in de praktijkstudie.
De info die verkregen wordt in deze literatuurstudie zal de basis leggen voor de volgende fase van dit onderzoek: de praktijkstudie.
Deze fase zou starten halverwege maart en loopt tot eind april met een duur van 6 weken. Deze praktijkstudie opgedeeld in 3 delen.
het ontwerpen van een virtuele omgeving, de implementatie van het centraal truststorebeheer en de evaluatie van de oplossing.
In de eerste stap zal er weer worden gekeken naar veel gebruikte software en besturingssystemen binnen bedrijven om zo een realistische infrastructuur te creëren.
Ook wordt er een certificate authority (CA) opgezet om het beheer van certificaten en een interne private PKI te simuleren.
Deze virtuele omgeving zal worden opgezet binnen GNS3, een netwerkemulator die het mogelijk maakt om complexe netwerken te simuleren.
De tweede stap, de implementatie van centraal truststorebeheer, begint eind maart en duurt drie weken.

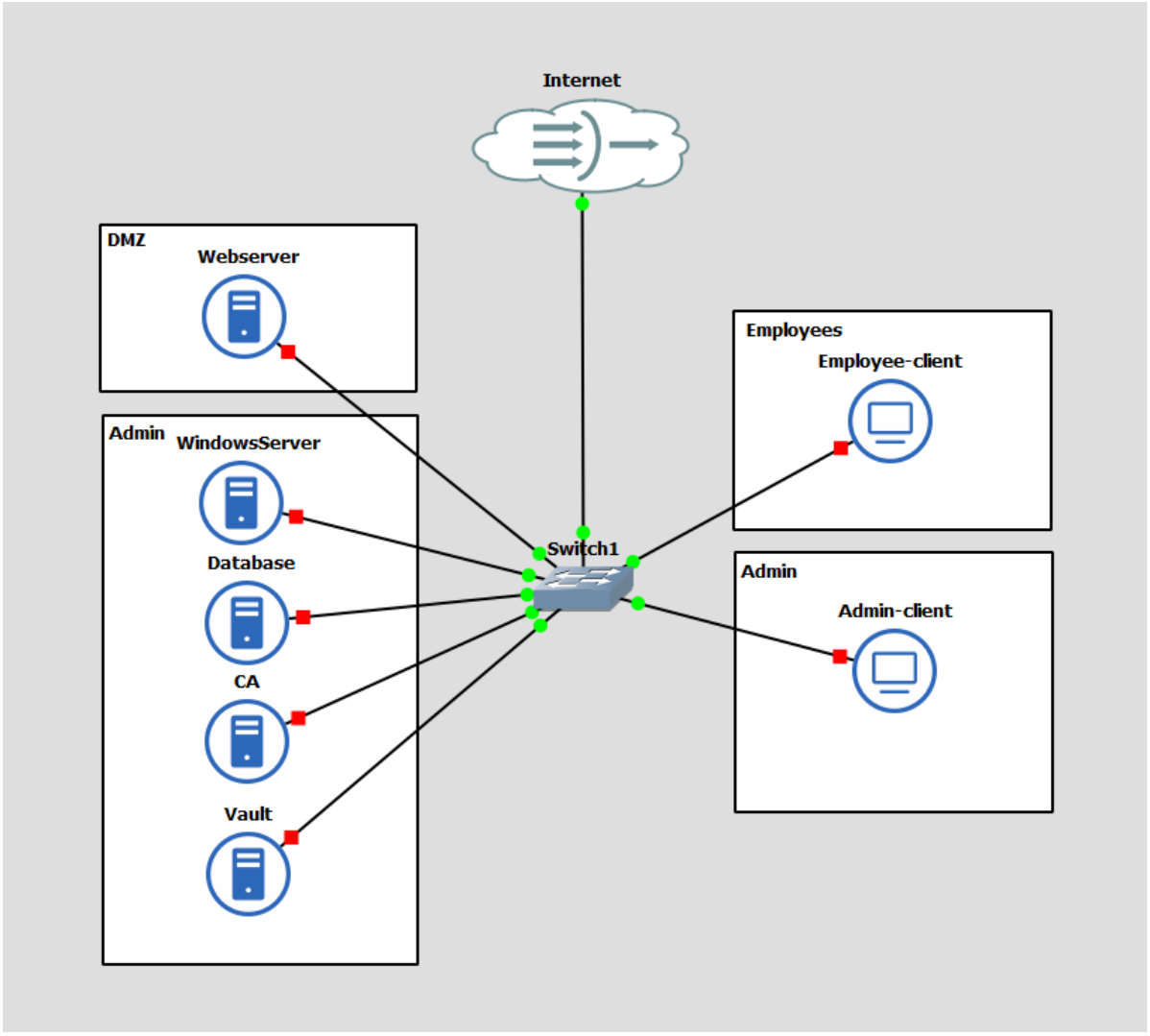
In deze fase wordt een gecentraliseerde oplossing voor truststorebeheer opgezet. Dit gebeurt door middel van de tools die gekozen werden samen met de co-promotor in de literatuurstudie in combinatie met eventuele zelfgeschreven scripts voor het certificaatbeheer.
Het doel hier is dat bij het genereren of updaten van een nieuw root certificaat, de truststores op elk systeem in het netwerk en zijn segmenten consistent zijn. Ook zullen er veiligheidscontroles geïmplementeerd worden, zoals het monitoren voor ongeldige of verlopen certificaten.
Na de implementatie volgt de derde stap van de praktijkstudie, die bestaat uit de evaluatie en validatie van de oplossing.
Deze fase begint begin halfweg april en duurt 2 weken.
De centrale vraag is hoe kwaliteitsvol de oplossing is en of deze voldoet aan de verwachtingen.
Er worden testcases opgesteld om veelvoorkomende uitdagingen te simuleren, zoals het herroepen van een rootcertificaat of het omgaan met verlopen certificaten. Ook wordt de schaalbaarheid en efficiëntie van de oplossing getest.
Bij de schaalbaarheid zal er gekeken worden naar hoe de oplossing omgaat met het veranderen of uitbreiden van het netwerk. Bij de efficiëntie zal er gekeken worden naar hoe snel aanpassingen van certificaten binnen de centrale truststore verspreid worden binnen het netwerk zonder het verstoren van andere diensten.
Als laatste fase van dit onderzoek zal er een rapportage gemaakt worden. Dit zal gebeuren in het begin van mei met een tijdsduur van 4 weken. De rapportage zal de ontwerpkeuzes en oplossing binenn de PoC beschrijven alsook de evaluatie en resultaten van deze oplossing.
Op basis van de bevindingen worden dan conclusies en aanbevelingen geformuleerd voor de implementatie van centraal truststorebeheer alsook welke aspecten mogelijks verder onderzoek vereisen.

4

Proof-of-concept

4.1. Virtuele omgeving

Voor het opzetten van een oplossing en deze te kunnen testen en evalueren, is er een virtuele omgeving opgezet. Deze virtuele omgeving is opgezet in GNS3, een netwerkemulator die het mogelijk maakt om netwerkapparatuur alsook virtuele machines te simuleren. Binnen deze virtuele omgeving zal er aan netwerksegmentatie gedaan worden, het netwerk zal opgedeelde worden in 3 segmenten: een Admin, DMZ en employee segment. Binnen het Admin segment zullen grotendeels van de servers staan alsook een workstation, in het DMZ segment zal er 1 server staan en in het employee segment zullen de werkstations staan. Deze segmentatie bestaat met als doel om het aantal vertrouwde root certificaten te beperken tot de benodigdheden van de verschillende segmenten, zo worden er minder root certificaten vertrouwd door de verschillende segmenten wat tot lagere security risico's leidt. Binnen het Admin segment staan er 4 servers, "Windowsserver", "Database", "CA" en "Vault" alsook is hier een workstation "Admin-client". Het DMZ segment heeft een server "Webserver" en het employee segment heeft 1 workstation "Employee-client".



4.1.1. Webserver

De webserver draait op Ubuntu 22.04 en heeft een Nginx webserver draaien met een certificaat dat ondertekend werd door de CA server van binnen deze virtuele omgeving. Het doel van deze webserver is om de functionaliteit van het aanpassen van de truststore te testen, als een end-point de webpagina kan bereiken zonder waarschuwing via het https protocol, dan betekent dat de server ”CA” zijn root certificaat aanwezig is in de end-point zijn truststore.

4.1.2. Windows server

Deze server draait Windows Server 2022 en is de domein controller van de virtuele omgeving. Het domein is ”Bachelorproef.local” en de server heeft een Active Directory (AD) opgezet. De Active Directory bevat 2 logins voor de Windows clients ”Employee1” en ”Admin1”. Binnen de AD zal elke Windows client die het netwerk betreedt automatisch gestoken worden. Deze clients worden manueel in de juiste organisational unit (OU) gestoken. In dit geval zijn dit de OU ”Employee” en ”Ad-

min".

4.1.3. CA

De CA server draait een Almalinux 8.8 image en is een Certificate Authority (CA). Het root certificaat is gemaakt met OpenSSL en is ondertekend door de CA zelf. Zoals eerder vermeld is er een leaf certificaat aangemaakt en ondertekend voor de webserver.

4.1.4. Vault

De Vault server draait ook Ubuntu 22.04 en heeft een Hashicorp Vault server draaien. Deze vault server zal gebruikt worden om de root certificaten op een centrale plaats op te slaan.

4.1.5. Employee-client

Deze Windows client draait Windows 11, en bevindt zich in het Employee netwerksegment. Deze client maakt deel uit van het domein "Bachelorproef.local" en is ook lid van de OU "Employee".

4.1.6. Admin-client

Deze Windows client draait ook Windows 11 en bevindt zich in het Admin netwerksegment. Deze client maakt ook deel uit van het domein "Bachelorproef.local" en is ook lid van de OU "Admin".

4.2. Eerste oplossing

4.2.1. Oplossing voor Windows end-points door middel van GPO's met root certificaten

Om Windows clients hun truststores centraal te kunnen beheren, kan er simpelweg een GPO (Group Policy Object) aangemaakt worden die root certificaten bevat en deze kan dan toegepast worden op een OU binnen de Active Directory. Er kan dus een GPO aangemaakt worden per OU (in dit geval de 3 OU's voor de 3 netwerksegmenten) die elk de juiste root certificaten bevatten. Voor deze proof of concept zal er een verzameling van root certificaten opgeslagen worden in een directory op de Windows server, deze directory zal 3 subdirectories bevatten voor elk netwerksegment. De root certificaten die in deze subdirectories staan zijn de root certificaten die de clients in dat netwerksegment moeten vertrouwen. De GPO's zullen dan de root certificaten uit deze subdirectories halen en deze toevoegen aan de truststore van de clients in dat netwerksegment. De GPO's kunnen als volgt aangemaakt worden:

- Open de Group Policy Management Console (GPMC) op de Windows server.

- Maak een nieuwe GPO aan door met de rechtermuisknop op de OU te klikken en "Create a GPO in this domain, and Link it here" te selecteren.
- Geef de GPO een naam, bijvoorbeeld "Employee-certificates".
- Klik met de rechtermuisknop op de GPO en selecteer "Edit".
- Ga naar Computer Configuration > Policies > Windows Settings > Security Settings > Public Key Policies > Trusted Root Certification Authorities.
- Klik met de rechtermuisknop en selecteer "Import".
- Volg de wizard om het root certificaat te importeren vanuit de directory op de Windows server.
- Herhaal deze stappen voor de andere OU's en root certificaten.

De GPO's kunnen dan toegepast worden, dit kan door met de rechtermuisknop op de OU te klikken en "Link an Existing GPO" te selecteren. Selecteer dan de GPO die je wilt toepassen op de OU.

Op de clients kan je dan de GPO's toepassen door het volgende commando uit te voeren in de command prompt:

```
gpupdate /force
```

Ook kan de GPO toegepast worden door de clients opnieuw op te starten.

4.2.2. Oplossing voor Linux end-points met Ansible

Om het probleem van het centraal beheren van de root certificaten op te lossen, kan er een Ansible playbook gemaakt worden dat root certificaten kan toevoegen aan de truststores van de Linux systemen. De CA server zal in deze proof-of-concept ook de Ansible control server zijn. Dit is de server die de Ansible playbooks zal uitvoeren op de end-points. Hiervoor is een inventory bestand nodig dat de IP adressen van de end-points bevat alsook de locatie van de key en gebruiker die gebruikt zal worden voor de SSH verbinding. Dit bestand voor deze omgeving ziet er als volgt uit:

```
[DMZ]
web ansible\_host=bpweb.bachelorproef.local ansible\_user=ubuntu ansi-
↪ ble\_ssh\_private\_key\_file=/home/almalinux/.ssh/id\_rsa

```

```
[Admin]
db ansible\_host=db.bachelorproef.local ansible\_user=almalinux ansi-
ble\_ssh\_private\_key\_file=/home/almalinux/.ssh/id\_rsa
ca ansible\_host=ca.bachelorproef.local ansible\_user=almalinux ansi-
ble\_ssh\_private\_key\_file=/home/almalinux/.ssh/id\_rsa
```

Net zoals bij de Windows server zal er een directory aangemaakt worden op de server "CA" die subdirectories bevat voor elk netwerksegment. Deze subdirectories bevatten de root certificaten die de end-points in dat netwerksegment moeten vertrouwen. Met deze bestanden en directories kan er een Ansible playbook gemaakt worden dat de root certificaten uit de juiste subdirectory haalt en toevoegt aan de truststore van de end-points. Dit kan gedaan worden met het volgende Ansible playbook:

```
- name: Installeer en vervang alle root certificaten per groep
    hosts: all
    become: true
    vars:
    cert\_source\_dir: >-
        {{ '/home/almalinux/Ansible/DMZ' if 'DMZ' in group\_names else '/home/almalinux/Ansible/Admin' }}
    dest\_cert\_dir: >-
        {{ '/usr/local/share/ca-certificates' if ansible\_os\_family == "Debian" else '/etc/pki/ca-trust/source/anchors' }}
    update\_command: >-
        {{ 'update-ca-certificates' if ansible\_os\_family == "Debian" else 'update-ca-trust extract' }}

    tasks:

    ### BACKUP STANDAARD TRUSTSTORE ###
    - name: Maak back-up van standaard truststore op Debian
        when: ansible\_os\_family == "Debian"
        command: >
        tar czf /root/debian-ca-certificates-backup.tar.gz /usr/share/ca-certificates /etc/ssl/certs

    - name: Maak back-up van standaard truststore op RedHat
        when: ansible\_os\_family == "RedHat"
```

```
22          command: >
23          tar czf /root/redhat-ca-certificates-backup.tar.gz
             ↪ /etc/pki/ca-trust
24
25      ### VERWIJDER STANDAARD CERTIFICATEN ###
26      - name: Ledig standaart truststore op Debian
27          when: ansible\_os\_family == "Debian"
28          file:
29          path: "{{ item }}"
30          state: absent
31          loop:
32          - /usr/share/ca-certificates/mozilla
33          - /etc/ssl/certs/ca-certificates.crt
34
35      - name: Ledig standaart truststore op RedHat
36          when: ansible\_os\_family == "RedHat"
37          shell: rm -f /etc/pki/ca-trust/source/anchors/*
38          args:
39          warn: false
40
41      ### CERTIFICATEN INSTALLEREN ###
42      - name: Zoek alle certificaten in groepsspecifieke map
43          find:
44          paths: "{{ cert\_source\_dir }}"
45          patterns: "*.crt"
46          register: certs\_to\_install
47
48      - name: Kopieer certificaten naar truststore directory
49          copy:
50          src: "{{ item.path }}"
51          dest: "{{ dest\_cert\_dir }}/{{ item.path | basename }}"
52          owner: root
53          group: root
54          mode: '0644'
55          loop: "{{ certs\_to\_install.files }}"
56          when: certs\_to\_install.matched > 0
57
58      ### TRUST STORE UPDATEN ###
59      - name: Update systeem truststore
60          command: "{{ update\_command }}"
61
62      ### VERIFICATIE ###
```

```
63          - name: Bevestiging
64              debug:
65              msg: "Alle root certificaten zijn geïnstalleerd en de
                ↪  truststore is bijgewerkt!"
```

De playbook begint met het maken van een back-up van de standaard out-of-the-box truststores, dit is belangrijk aangezien de tweede stap deze zal leegmaken. Moest een belangrijke root certificaat ontbreken na het uitvoeren van de playbook, kan deze teruggezet worden met de back-up. Deze playbook kijkt vervolgens naar welke groep de end-point inzit om zo de certificaten uit de directory met dezelfde naam te halen, daarna worden de certificaten gekopieerd naar de directory op de end-points die de truststores bekijken en tenslotte wordt de truststore geüpdatet met het juiste commando. Deze playbook bepaalt de bestemmingsdirectory en het update commando op basis van de 'OS family' van de end-point. In dit geval is dit Debian of Red Hat, voor andere OS families zouden deze nog manueel moeten worden toegevoegd samen met de juiste commando's en directories.

Om het voorgaande Ansible playbook uit te voeren, kan er gebruik gemaakt worden van de volgende commando's:

```
1       ansible-playbook -i inventory playbook.yml
```

Waarbij "inventory" het inventory bestand is dat de IP adressen van de end-points bevat en "playbook.yml" het Ansible playbook is dat de root certificaten toevoegt aan de truststores van de end-points.

4.2.3. Pro's en Con's van de eerste oplossing

- Pro's:
  - De root certificaten kunnen eenvoudig beheerd worden.
  - Geen nodige installatie van extra software op de end-points.
  - End-points ondervinden geen hinder van de installatie van de root certificaten.
  - Er zijn weinig points of failure aangezien de oplossing geen bijkomende servers of software nodig heeft.
- Con's:
  - De GPO's passen alleen maar toe wanneer de end-points opnieuw opgestart worden of de GPO's geforceerd worden.

- Het importeren van root certificaten in de GPO's kan niet geautomatiseerd worden, dit betekent dat als er veel root certificaten zijn, het handmatig aanmaken van de GPO's tijdrovend zal zijn.
- De certificaten staan in directories die niet beveiligd zijn, dit brengt een risico mee dat de certificaten binnen de directory kunnen worden aangepast of verwijderd.
- Zowel de Windows server als de CA server (de Ansible control server) hebben een kopie van de root certificaten, bij een aanpassing van de root certificaten moeten deze op beide servers aangepast worden.
- De Ansible playbook is niet schaalbaar, een server maakt SSH verbindingen met de end-points en voert de playbook uit op elke end-point. Dit betekent dat als er veel end-points zijn, de Ansible playbook traag zal zijn en mogelijk niet alle end-points kan bereiken.

4.3. Tweede oplossing

4.3.1. Installeren van een Vault server

Om het probleem van het dubbel beheren van de root certificaten op te lossen, moet er een centraal punt op het netwerk zijn die de root certificaten bevat. Dit kan gedaan worden door een Hashicorp Vault server op te zetten. Deze server brengt naast het beheren van de root certificaten ook autorisatie met zich mee waardoor de certificaten ook beveiligd zijn. Binnen deze proof-of-concept omgeving is er een standalone server voorzien voor Hashicorp Vault genaamd "Vault". Deze server heeft een kopie van de directories met root certificaten die op de server "CA" stond. Daarnaast zal Vault in "dev" mode draaien, dit is een testmodus die het mogelijk maakt om snel te experimenteren met Vault zonder dat er een volledige configuratie nodig is. In deze modus zal de vault server ook geen persistentie hebben, wat betekent dat alle gegevens uit de vault verloren gaan als de vault sluit of de server stopt.
De vault server wordt opgestart met de volgende commando:

```
1    vault server -dev -dev-listen-address="0.0.0.0:8200"
```

Dit maakt de Vault server toegankelijk op alle IP adressen van de server op poort 8200.
De Vault server zal onderverdelingen hebben per netwerksegment, binnen elk netwerksegment zullen de root certificaten staan. Naast de root certificaten zal er ook een manifest binnen elk netwerksegment bestaan die een oplijsting bevat van de root certificaten die binnen dat netwerksegment staan. Deze manifest kan gebruikt worden door end-points om elk root certificaat binnen een bepaald netwerksegment op te halen. Voor het aanmaken van de indelingen en manifests werd er

een bash script gemaakt die de root certificaten uit de directories haalt en deze toevoegt aan de vault. Dit script ziet er als volgt uit:

```
#!/bin/bash

VAULT\_ADDR="http://127.0.0.1:8200"
export VAULT\_ADDR

BASE\_CERT\_DIR="/home/ubuntu/"
SEGMENTS=("DMZ" "Admin" "Employee")

for SEGMENT in "${SEGMENTS[@]}"; do
        SEGMENT\_DIR="$BASE\_CERT\_DIR/$SEGMENT"
        VAULT\_PATH="secret/certs/$SEGMENT"

        echo "Uploading certificates for segment: $SEGMENT"

        for cert\_file in "$SEGMENT\_DIR"/*.pem
         ↪ "$SEGMENT\_DIR"/*.crt; do
                [ -e "$cert\_file" ] || continue  # Skip if no
                 ↪ matching files
                cert\_name=$(basename "$cert\_file" | sed
                 ↪ 's/\.[^.]*$//')  # remove extension

                echo " - Uploading $cert\_name from $cert\_file"

                vault kv put "$VAULT\_PATH/$cert\_name"
                 ↪ cert=@"$cert\_file"
        done
        # Create a manifest (list of certs) for segment-based
         ↪ fetching
        echo "Generating manifest for $SEGMENT"
        cert\_keys=$(ls "$SEGMENT\_DIR" | sed 's/\.[^.]*$//' | jq
         ↪ -R . | jq -s .)
        vault kv put "$VAULT\_PATH/\_manifest" keys="$cert\_keys"

done

echo "All certificates uploaded to Vault."

```

Voor de beveiliging van de root certificaten in de vault zal er gebruik gemaakt wor-

den van token authenticatie. Dit is een manier van authenticatie waarbij een token door de end-points wordt gebruikt om toegang te krijgen tot de vault. Om ervoor te zorgen dat de rechten met deze token gelimiteerd zijn, kan er een policy aangemaakt worden. De volgende policy kan gebruikt worden voor read-only toegang tot alle root certificaten op alle netwerksegmenten:

```
path "secret/data/certs/*" {
capabilities = ["read", "list"]
}

path "secret/metadata/certs/*" {
capabilities = ["read", "list"]
}

path "secret/data/certs/*/manifest" {
capabilities = ["read"]
}

path "secret/data/certs/*/*" {
capabilities = ["read"]
}
```

Deze policy kan dan toegevoegd worden aan een role en uiteindelijk kan een token aangemaakt worden met deze role. Eerst moet approle, de manier van authenticatie aangezet worden met het volgende commando:

```
vault auth enable approle
```

Daarna kunnen we de voorgaande policy aanmaken met het volgende commando:

```
vault policy write certs-read public-certs.hcl
```

Waarbij "public-certs.hcl" het bestand is dat de policy bevat en "certs-read" de naam die gegeven wordt aan de policy. De role kan dan aangemaakt worden als volgt:

```
vault write auth/approle/role/chef\_cert\_reader \
token\_policies="certs-read" \
secret\_id\_ttl="60m" \
token\_ttl="30m" \
token\_max\_ttl="60m"
```

Waarbij "chef_cert_reader" de naam is die gegeven wordt aan de role. Token_policies linkt de policy die zojuist werd gemaakt aan de role, secret_id_ttl is de tijd dat de secret_id geldig is, token_ttl is de tijd dat de token geldig is en token_max_ttl is de maximale tijd dat de token geldig is. Clients kunnen via de role_id en secret_id een token aanvragen, voor het bekomen van de role_id kan er gebruik gemaakt worden van het volgende commando:

```
1    vault read auth/approle/role/chef\_cert\_reader
```

Deze role_id blijft geldig tot en met de vault server opnieuw opgestart wordt. De secret_id is ook nodig, deze kan aangemaakt worden met het volgende commando:

```
1    vault write -f auth/approle/role/chef\_cert\_reader/secret-id
```

De secret_id heeft een bepaalde levensduur die eerder werd ingesteld bij het maken van de role.

4.3.2. Installeren van een Vault agent

Voor de clients om toegang te krijgen tot de vault, moet er een token aangemaakt worden met de role_id en secret_id waarover hiervoor werd gesproken. Om deze token aan te maken, moet er een vault agent draaien op de end-point. Deze agent kan geconfigureerd worden met het volgende bestand '/etc/vault/agent-config.hcl':

```
1    pid\_file = "/run/vault/vault-agent.pid"
2
3    vault {
4    address = "http://vault.bachelorproef.local:8200"
5    }
6
7    auto\_auth {
8    method "approle" {
9        mount\_path = "auth/approle"  # default path unless you
          ↪  mounted AppRole elsewhere
10       config = {
11       role\_id\_file\_path    = "/var/lib/vault/role\_id"
12       secret\_id\_file\_path = "/var/lib/vault/secret\_id"
13       remove\_secret\_id\_file\_after\_reading = false
14       }
15   }
16
```

```
    sink "file" {
        config = {
        path = "/var/lib/vault/agent-token"
        }
    }
    }

    cache {
    use\_auto\_auth\_token = true
    }

    listener "tcp" {
    address     = "0.0.0.0:8201"
    tls\_disable = true
    }
```

Waarbij "role_id_file_path" het pad is naar het bestand dat de role_id bevat en "secret_id_file_path" het pad is naar het bestand dat de secret_id bevat. Dit bestand kan dan opgeslagen worden op de end-point in de directory "/var/lib/vault/". Sink "file" zorgt ervoor dat de token die aangemaakt wordt door de agent, opgeslagen wordt in het bestand "/var/lib/vault/agent-token". Dit bestand kan dan gebruikt worden door de end-point om toegang te krijgen tot de vault.

Om de agent te starten kan er gebruik gemaakt worden van het volgende commando:

```
    vault agent -config=/etc/vault/agent-config.hcl
```

De agent zal dan automatisch de token aanmaken en hernieuwen wanneer deze verloopt.

4.3.3. Oplossing voor Windows end-points door middel van GPO's met startup scripts en SCCM

Aangezien het importeren van root certificaten in de GPO's niet geautomatiseerd kan worden, is er geen schaalbaarheid in de vorige oplossing. Dit betekent dat als er veel root certificaten zijn, het handmatig aanmaken van de GPO's een grote kost in tijd zal zijn. Om dit probleem op te lossen kan er in de plaats van root certificaten aan de GPO's te koppelen, een startup script gemaakt worden dat de root certificaten ophaalt en deze toevoegt aan de truststore van de clients. Hiervoor moet een PowerShell script geschreven worden dat de root certificaten ophaalt uit de vault van het juiste netwerksegment en deze toevoegt aan de truststore van de end-points. Dit kan gedaan worden met het volgende PowerShell script:

```
# Vault server address and token
$vaultServer = "http://vault.bachelorproef.local:8200"
$vaultToken = "s.xxxxxxx" # de token die voor de client werd aangemaakt
$networkSegment = "DMZ" # het netwerksegment van de client
```

Dit script kan dan toegevoegd worden aan de GPO's die gelinkt staan aan de OU's. Dit kan gedaan worden door met de rechtermuisknop op de GPO te klikken en "Edit" te selecteren. Ga dan naar Computer Configuration > Policies > Windows Settings > Scripts (Startup/Shutdown) > Startup. Klik met de rechtermuisknop en selecteer "Properties". Voeg het PowerShell script toe aan de startup scripts.

Om ervoor te zorgen dat het script niet alleen bij de opstart van de client wordt uitgevoerd, maar ook achter een bepaalde interval, kan er gebruik gemaakt worden van SCCM (System Center Configuration Manager). Dit is een tool die het mogelijk maakt om software en updates te beheren op de Windows clients. De task sequence kan gemaakt worden met het volgende PowerShell script:

```
# SCCM server address and credentials
$sccmServer = "sccm.bachelorproef.local"
$sccmUsername = "admin"
$sccmPassword = "password"

# Create a new SCCM task sequence
$taskSequenceName = "Update GPOs"
$taskSequenceDescription = "Update GPOs on all clients"
$taskSequence = New-SCCMTaskSequence -Name $taskSequenceName -Description $taskSequenceDescription -Server $sccmServer -Username $sccmUsername -Password $sccmPassword

# Add the GPO update step to the task sequence
Add-SCCMTaskSequenceStep -TaskSequence $taskSequence -StepType "Run Command" -Command "gpupdate /force" -Server $sccmServer -Username $sccmUsername -Password $sccmPassword

# Deploy the task sequence to all clients
Deploy-SCCMTaskSequence -TaskSequence $taskSequence -CollectionName "All Systems" -Server $sccmServer -Username $sccmUsername -Password $sccmPassword
```

4.3.4. Oplossing voor Linux end-points met Chef en Vault

Een probleem met de vorige oplossing is dat Ansible een lage schaalbaarheid heeft. Dit betekent dat als er veel end-points zijn, de Ansible playbook traag zal zijn en mogelijk niet alle end-points kan bereiken. Om dit probleem aan te pakken kan er gebruikt gemaakt worden van Chef. Chef verschilt van Ansible op het vlak van wie de instructies uitvoert. Bij Ansible is de Ansible control server verantwoordelijk voor het uitvoeren van de instructies op de end-points, terwijl bij Chef de end-points zelf verantwoordelijk zijn voor het uitvoeren van de instructies. Hiervoor is er beide nood aan een Chef infra server en een Chef infra client. De infra server is de server die de instructies zal geven aan de end-points en de infra client is de end-point die de instructies zal uitvoeren op de end-points.

De infra server kan geconfigureerd worden met het volgende commando:

```
1    chef-server-ctl reconfigure
```

Dit zal prompten om de naam van de server, de organisatie en de naam van de validator. Dit zijn de gegevens die gebruikt worden om de infra server te configureren. Voor deze proof-of-concept zal de infra server draaien op de server ”CA” en zullen de naam, organisatie en validator respectievelijk ”ca”, ”bachproef” en ”bachproef-validator” zijn.

Chef maakt gebruikt van cookbooks, dit zijn bestanden die de instructies bevatten voor de end-points. Om cookbooks te maken en te uploaden naar de chef infra server, moet er een chef workstation geïnstalleerd worden. Binnen deze omgeving bevindt deze zich op de server ”Database” aangezien deze server momenteel geen andere taken heeft. Chef workstation kan gebruikt worden met het ’knife’ commando, dit is een command line tool die gebruikt wordt om te communiceren met de chef infra server. Chef workstation wordt geconfigureerd met het volgende bestand ’knife.rb’:

```
1    current\_dir = File.dirname(\_\_FILE\_\_)
2    log\_level                :info
3    log\_location             STDOUT
4    node\_name                'db.bachelorproef.local'
5    client\_key               ”/home/almalinux/.chef/bpchef.pem”
6    chef\_server\_url
      ↪  'https://ca.bachelorproef.local/organizations/bachproef'
7    cookbook\_path            [”/home/almalinux/cookbooks/”]
8    ssl\_verify\_mode :verify\_none
9    editor 'vi'
```

Ssl_verify_mode is ingesteld op ”verify_none” wegens dat dit een testomgeving is. In een productieomgeving zou dit ingesteld moeten worden op ”verify_peer” waarbij het certificaat van de chef infra server gevalideerd wordt.
Eenmaal de chef workstation is geconfigureerd kunnen er cookbooks gemaakt worden. Zoals in het configuratiebestand te zien is, worden de cookbooks opgeslagen in de directory ”/home/almalinux/cookbooks/”. Voor het oplossen van het probleem in dit onderzoek zal er een cookbook gemaakt worden dat de root certificaten uit de vault haalt en deze toevoegt aan de truststore van de end-points. Elke cookbook moet zijn eigen subdirectory hebben binnen de cookbooks directory. In dit geval werd de cookbook ’truststore_update’ aangemaakt in de directory ”/home/almalinux/cookbooks/truststore_update/”. Elke cookbook moet een metadata.rb bestand hebben dat de naam van de cookbook bevat. Dit bestand ziet er als volgt uit:

```
1   cookbooks/truststore\_update/metadata.rb
```

Naast de metadata.rb moet er een recipe directory zijn die de instructies van de cookbook bevat en kan eventueel een attributes directory gemaakt worden die default waarden voor attributen gedefinieerd heeft. De cookbook die voor deze oplossing zal gebruikt worden is te vinden in de directory ”/home/almalinux/cookbooks/truststore_update/recipes/default.rb” en ziet er als volgt uit:

```
1   require 'vault'
2   require 'json'
3
4   Vault.address = 'http://vault.bachelorproef.local:8200'
5   Vault.token = ::File.read(”/var/lib/vault/agent-token”).strip
6
7   network\_segment = node['network\_segment'] || raise(”Missing
     ↪  network\_segment attribute”)
8
9   # Pull the manifest
10  Chef::Log.info(”Fetching manifest for network segment:
     ↪  #{network\_segment}”)
11  manifest\_secret =
     ↪  Vault.logical.read(”secret/data/certs/#{network\_segment}/\_manifest”)
12  manifest\_data = manifest\_secret.data[:data]
13
14  Chef::Log.info(”Manifest data: #{manifest\_data.inspect}”)
15
16  # Retrieve the 'keys' field from manifest\_data
```

```
keys = manifest\_data[:keys] # Use symbolized key

# Log the raw keys value for debugging
Chef::Log.info("Raw 'keys' value: #{keys.inspect}")

# Validate the 'keys' field
if keys.nil? || keys.strip.empty?
  raise "No 'keys' field found in manifest or 'keys' is empty"
end

# Parse the 'keys' field as JSON
begin
  Chef::Log.info("Parsing 'keys' field as JSON...")
  cert\_list = JSON.parse(keys.strip) # Directly parse the keys
   ↪ string as JSON after stripping whitespace
rescue JSON::ParserError => e
  raise "Failed to parse 'keys' as JSON: #{e.message}"
end

# Log the parsed cert\_list for debugging
Chef::Log.info("Parsed cert\_list: #{cert\_list.inspect}")

# If cert\_list is empty, raise an error
if cert\_list.empty?
  raise "No certificates found in manifest"
end

# Process each certificate
cert\_list.each do |cert\_name|
  Chef::Log.info("Processing certificate: #{cert\_name}")
  cert\_path =
   ↪ "secret/data/certs/#{network\_segment}/#{cert\_name}"
 Chef::Log.info("Fetching certificate from path: #{cert\_path}")

  cert\_secret = Vault.logical.read(cert\_path)
  Chef::Log.info("Vault response for #{cert\_path}:
   ↪ #{cert\_secret.inspect}")

  cert\_pem = cert\_secret.data[:data][:cert] # Corrected to use
   ↪ :cert as the key
  Chef::Log.info("Certificate content for #{cert\_name}:
   ↪ #{cert\_pem.inspect}")
```

```
54
55      # Ensure cert\_pem is not nil or empty
56      if cert\_pem.nil? || cert\_pem.strip.empty?
57        raise "Certificate content for #{cert\_name} is empty or
           ↪  missing"
58      end
59
60      target\_path = case node['platform\_family']
61                    when 'debian'
62                      "/usr/local/share/ca-
                         ↪  certificates/#{cert\_name}.crt"
63                    when 'rhel'
64                      "/etc/pki/ca-
                         ↪  trust/source/anchors/#{cert\_name}.crt"
65                    else
66                      raise "Unsupported platform family"
67                    end
68
69      file target\_path do
70        content cert\_pem
71        owner 'root'
72        group 'root'
73        mode '0644'
74        notifies :run, 'execute[update-trust-store]', :delayed
75      end
76    end
77
78    # Update trust store
79    execute 'update-trust-store' do
80      command case node['platform\_family']
81              when 'debian'
82                'update-ca-certificates'
83              when 'rhel'
84                'update-ca-trust extract'
85              else
86                raise "Unsupported platform family"
87              end
88      action :nothing
89    end
```

De directory attributes bevat een default.rb bestand dat er als volgt uitziet:

```
default['platform\_family'] = 'rhel'
```

De reden voor deze default waarde is omdat de RHEL image die gebruikt werd incorrect de platform_family teruggeeft.

Met alle nodige bestanden en directories kan de cookbook worden geüpload naar de chef infra server met het volgende commando:

```
knife cookbook upload truststore\_update
```

End-points hebben dan een chef client nodig om de cookbooks te kunnen opvragen en uitvoeren. Deze kan geïnstalleerd worden vanuit de chef infra server met het volgende commando:

```
knife bootstrap <ip\_address> \
--ssh-user <user> \
--sudo \
--use-sudo-password \
--node-name <node\_name> \
--run-list 'recipe[truststore\_update]' \
--json-attribute '{"network\_segment": "segment\_name"}'
```

Dit commando zou er als volgt uitzien voor de server "webserver" met hostname "bpweb":

```
knife bootstrap bpweb.bachelorproef.local \
--ssh-user ubuntu \
--sudo \
--use-sudo-password \
--node-name bpweb \
--run-list 'recipe[truststore\_update]' \
--json-attribute '{"network\_segment": "DMZ"}'
```

Elke Chef client moet dus een attribuut "network_segment" hebben, deze variabele wordt gebruikt voor de juiste subdirectory te kiezen in de vault. Daarnaast de run-list parameter ingevuld worden met de naam van de cookbook die eerder werd gemaakt.

Nu moeten de clients het cookbook uitvoeren, dit kan gedaan worden met het volgende commando:

```
chef-client
```

De client zal alle cookbooks ophalen van de chef infra server en deze uitvoeren. Dit kan ook geconfigureerd worden dat dit automatisch gebeurd op een bepaalde tijd, dit kan gedaan worden met het volgende commando:

```
echo "chef-client -i 1800" >> /etc/cron.d/chef-client
```

Dit zorgt ervoor dat de chef-client elke 30 minuten zal draaien en de cookbooks zal ophalen van de chef infra server. Met deze opstelling zouden de Linux end-points nu steeds up-to-date moeten zijn met de root certificaten in de vault.

4.3.5. Pro's en Con's van de tweede oplossing

- Pro's:
  - Root certificaten kunnen beheerd worden op een centrale plaats.
  - Met de SCCM task sequence blijven de Windows end-points up-to-date ook als ze niet opnieuw opgestart worden.
  - Alle workload wordt gedaan door de end-points zelf, wat zorgt voor hogere schaalbaarheid.
- Con's:
  - De oplossing is afhankelijk van verschillende servers, wat voor meer points of failure zorgt.
  - Er zijn meer servers en software nodig om te onderhouden.

4.4. Overwegingen en aanbevelingen

Deze proof-of-concept is een goede start voor het aantonen van de mogelijkheden tot het centraal beheren van root certificaten op een netwerk. Er zijn echter nog enkele overwegingen en aanbevelingen die gedaan kunnen worden om de oplossing te verbeteren.

4.4.1. Schaalbaarheid

In de praktijk kunnen beide oplossingen grotere schaalbaarheid hebben door de nodige bronnen (Vault, Ansible control server, Chef infra server, Active Directory) te voorzien in elk netwerksegment, hier kan dan onderzocht worden om de inhoud van de servers te repliceren naar de verschillende netwerksegmenten.

5

Conclusie

Curabitur nunc magna, posuere eget, venenatis eu, vehicula ac, velit. Aenean ornare, massa a accumsan pulvinar, quam lorem laoreet purus, eu sodales magna risus molestie lorem. Nunc erat velit, hendrerit quis, malesuada ut, aliquam vitae, wisi. Sed posuere. Suspendisse ipsum arcu, scelerisque nec, aliquam eu, molestie tincidunt, justo. Phasellus iaculis. Sed posuere lorem non ipsum. Pellentesque dapibus. Suspendisse quam libero, laoreet a, tincidunt eget, consequat at, est. Nullam ut lectus non enim consequat facilisis. Mauris leo. Quisque pede ligula, auctor vel, pellentesque vel, posuere id, turpis. Cras ipsum sem, cursus et, facilisis ut, tempus euismod, quam. Suspendisse tristique dolor eu orci. Mauris mattis. Aenean semper. Vivamus tortor magna, facilisis id, varius mattis, hendrerit in, justo. Integer purus.

Vivamus adipiscing. Curabitur imperdiet tempus turpis. Vivamus sapien dolor, congue venenatis, euismod eget, porta rhoncus, magna. Proin condimentum pretium enim. Fusce fringilla, libero et venenatis facilisis, eros enim cursus arcu, vitae facilisis odio augue vitae orci. Aliquam varius nibh ut odio. Sed condimentum condimentum nunc. Pellentesque eget massa. Pellentesque quis mauris. Donec ut ligula ac pede pulvinar lobortis. Pellentesque euismod. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos hymenaeos. Praesent elit. Ut laoreet ornare est. Phasellus gravida vulputate nulla. Donec sit amet arcu ut sem tempor malesuada. Praesent hendrerit augue in urna. Proin enim ante, ornare vel, consequat ut, blandit in, justo. Donec felis elit, dignissim sed, sagittis ut, ullamcorper a, nulla. Aenean pharetra vulputate odio.

Quisque enim. Proin velit neque, tristique eu, eleifend eget, vestibulum nec, lacus. Vivamus odio. Duis odio urna, vehicula in, elementum aliquam, aliquet laoreet, tellus. Sed velit. Sed vel mi ac elit aliquet interdum. Etiam sapien neque, convallis et, aliquet vel, auctor non, arcu. Aliquam suscipit aliquam lectus. Proin tincidunt magna sed wisi. Integer blandit lacus ut lorem. Sed luctus justo sed enim.

Morbi malesuada hendrerit dui. Nunc mauris leo, dapibus sit amet, vestibulum et, commodo id, est. Pellentesque purus. Pellentesque tristique, nunc ac pulvinar adipiscing, justo eros consequat lectus, sit amet posuere lectus neque vel augue. Cras consectetuer libero ac eros. Ut eget massa. Fusce sit amet enim eleifend sem dictum auctor. In eget risus luctus wisi convallis pulvinar. Vivamus sapien risus, tempor in, viverra in, aliquet pellentesque, eros. Aliquam euismod libero a sem. Nunc velit augue, scelerisque dignissim, lobortis et, aliquam in, risus. In eu eros. Vestibulum ante ipsum primis in faucibus orci luctus et ultrices posuere cubilia Curae; Curabitur vulputate elit viverra augue. Mauris fringilla, tortor sit amet malesuada mollis, sapien mi dapibus odio, ac imperdiet ligula enim eget nisl. Quisque vitae pede a pede aliquet suscipit. Phasellus tellus pede, viverra vestibulum, gravida id, laoreet in, justo. Cum sociis natoque penatibus et magnis dis parturient montes, nascetur ridiculus mus. Integer commodo luctus lectus. Mauris justo. Duis varius eros. Sed quam. Cras lacus eros, rutrum eget, varius quis, convallis iaculis, velit. Mauris imperdiet, metus at tristique venenatis, purus neque pellentesque mauris, a ultrices elit lacus nec tortor. Class aptent taciti sociosqu ad litora torquent per conubia nostra, per inceptos hymenaeos. Praesent malesuada. Nam lacus lectus, auctor sit amet, malesuada vel, elementum eget, metus. Duis neque pede, facilisis eget, egestas elementum, nonummy id, neque.

A

Onderzoeksvoorstel

Het onderwerp van deze bachelorproef is gebaseerd op een onderzoeksvoorstel dat vooraf werd beoordeeld door de promotor. Dat voorstel is opgenomen in deze bijlage.

A.1. Inleiding

In de digitale wereld is vertrouwen in beveiligingssystemen cruciaal, vooral binnen de financiële sector. Banken hebben complexe netwerken die moeten worden beschermd tegen cyberbedreigingen. Trust management, het beheren van vertrouwde relaties aan de hand van digitale certificaten en public key infrastructures (PKI's) speelt hierin een belangrijke rol.
Daarbij is het effectief beheren van truststores, de verzameling van vertrouwde certificaten, essentieel om veilige gegevensuitwisseling en transacties te garanderen. In vele netwerken worden truststores verspreid beheerd, hun locatie is afhankelijk van de gebruikte applicatie of besturingssystemen. Dit leidt tot een gefragmenteerde aanpak, wat niet alleen inefficiënt is, maar ook de kans vergroot op verouderde certificaten en beveiligszwaktes.
Door een centrale aanpak te ontwikkelen voor truststorebeheer wordt er gestreefd naar meer consistentie, eenvoud en veiligheid in het beheer van certificaten. Dit onderzoek richt zich op het formuleren van richtlijnen en oplossingen die specifiek van toepassing op de behoeftes van de financiële sector.
Om dit doel te bereiken, beantwoordt dit onderzoek de volgende deelvragen:
1. Wat zijn de belangrijkste concepten en technologieën achter PKI en truststorebeheer?
2. Wat zijn de verschillen tussen gedistribueerd en gecentraliseerd truststorebeheer, en welke voordelen biedt een gecentraliseerde aanpak?
3. Welke tools en technieken worden momenteel gebruikt voor truststorebeheer,

en wat zijn hun beperkingen?
4. Welke specifieke uitdagingen en eisen gelden voor truststorebeheer in bankomgevingen?
5. Welke componenten moeten worden opgenomen in een virtueel bankennetwerk om een realistisch beeld te creëren?
6. Hoe kan een gecentraliseerde oplossing voor truststorebeheer worden ontworpen en geïmplementeerd in een bankennetwerk?
7. Hoe kan de veiligheid, efficiëntie en schaalbaarheid van de gecentraliseerde oplossing worden getest en geëvalueerd?
8. In hoeverre voldoet de gecentraliseerde oplossing aan de beveiligings- en complianceeisen van een bankomgeving?
9. Wat zijn de praktische voordelen en beperkingen van een gecentraliseerde truststorebeheeroplossing in vergelijking met een gedistribueerde aanpak?
10. Welke aanbevelingen kunnen worden gedaan voor banken die een centrale truststorebeheerstrategie willen implementeren?
Door deze vragen te beantwoorden, streeft het onderzoek ernaar richtlijnen en oplossingen te formuleren die aansluiten bij de specifieke behoeften van de financiële sector.

A.2. Literatuurstudie

Bij het begrijpen van trust management is het belangrijk om te weten wat een public key infrastructure (PKI) en digitale certificaten zijn.
Een digitaal certificaat is een file die verbonden is met een cryptografische key pair en het authenticeert de identiteit van een website, individu, organisatie, gebruiker of apparaat. Zulke certificaten worden soms ook public key certificates of identity certificates genoemd
Een certificaat bevat het onderwerp, wat dient als de identiteit samen met een digitale signatuur. Het doel van digitale certificaten is om de identiteit en encryptie te verzekeren van een website, individu, organisatie, apparaat, gebruiker of server. (Digicert, 2025)
Een PKI beveiligd data en bouwt vertrouwen via deze digitale certificaten, die uitgegeven worden door certificate autorities (CA's) aan bedrijven nadat zij een certificate signing request (CSR) hebben gemaakt. Eens een CSR is goedgekeurd en een certificaat is uitgegeven kan een bedrijf of individu zijn communicatie, web domeinen en/of documenten authenticeren en beveiligen zolang het certificaat is gemaakt voor dit doeleinde en geldig is. (Topping, 2023)
Naast PKI en digitale certificaten zijn truststores en keystores ook van belang bij trust management.
IBM (2023) definieert dat Trust stores en keystores cryptografische artefacten bevatten, met andere woorden certificaten en private keys. Deze artefacten worden dan gebruikt door protocollen zoals TLS. Een keystore bevat persoonlijke certificaten

samen met de overeenkomstige private keys die gebruikt worden om de eigenaar van het certificaat te identificeren.
Voor een TLS verbinding stelt een persoonlijk certificaat de identiteit van een endpoint voor, beide de client en de server hebben dit certificaat om elkaar te identificeren.
Een truststore bevat dan weer signer certificates die door de andere endpoints vertrouwd worden, deze certificates zijn ook gekend als "certificate autority certificates".
Deze signer certificate bevatten een public key die gebruikt wordt om een persoonlijk certificaat te valideren. Door het toevoegen van de signer certificate in de client zijn truststore, kan de client de server vertrouwen om een TLS verbinding te maken.
Vele software en besturingssystemen komen dan ook vooraf geïnstalleerd met hun eigen trust stores.
"Er zijn 4 grote organisaties die zulke trust stores beheren: 1. Microsoft root certificate program die gebruikt wordt voor Windows. 2. Apple root certificate program is gebruikt voor alle Mac apparaten. 3. De Mozilla root certificates program wordt gebruikt door Mozilla zelf en meeste Linux distributies. 4. Google root certificate program wordt gebruikt door Google chrome en andere Google applicaties." (Dwivedi, 2024)
Volgens Mervana (2024) is een van de uitdagingen voor het beheren van trust stores, dat in grote organisaties met heterogene netwerken vele trust stores gedecentraliseerd liggen waarbij er een nood is aan gecoördineerd management.
Dit kan opgelost worden door een gecentraliseerd management systeem, dit maakt het makkelijker om veranderingen te coördineren.
Andere bronnen raden zulke gecentraliseerde systemen sterk aan.
"Organisaties moeten overwegen de default trust stores af te wijzen. Ze zouden een eigen corporate-level trust store moeten maken aan de hand van certificate whitelisting om te bepalen welke certificaten hierin worden opgenomen." (Arampatzis, 2020)
Bij het opstellen van deze centrale truststore binnen de financiële sector brengt dit ook enkele uitdagingen met zich mee. Er zijn namelijk beveiligingseisen voor de infrastructuur van banken waaraan moet worden voldaan.
Daarbij blijkt dat tegen het einde van 2024 de cyber security binnen europese banken moeten voldoen aan de " Digital Operational Resilience Requirements" (DORA). (Gorobeț, 2024)
Naast de DORA eisen zijn er ook nog andere eisen die van toepassing zijn op de financiële sector zoals de "Payment Services Directive 2" (PSD2).
In het onderzoek van Gounari e.a. (2024) blijkt dat PSD2 de nieuwe vereisten binnen communicatie systemen baseert op SCA, de nieuwe requirement die specifieke technische standaarden introduceert zoals PSD2-compliant certificaten.
In het uitvoeren van dit onderzoek zal er dus moeten worden gekeken naar welke

richtlijnen en eisen van toepassing zijn voor het gecentraliseerde management systeem alsook naar hoe deze kunnen worden geïmplementeerd.
Je mag deze sectie nog verder onderverdelen in subsecties als dit de structuur van de tekst kan verduidelijken.

A.3. Methodologie

Het onderzoek naar centrale truststorebeheer in een banknetwerk zal opgedeeld worden in drie fases: een literatuurstudie, praktijkstudie en uiteindelijke rapportage en oplevering.
In de eerste maand zal de eerste fase van het onderzoek uitgevoerd worden, de literatuurstudie.
Het doel in deze fase is om een diepgaand begrip te krijgen over trust management, certificaatbeheer en de verschillende aanpakken bij truststorebeheer.
In deze literatuurstudie zal er gekeken worden naar de bestaande theorieën, technologieën en andere tools die momenteel worden gebruikt in praktijk.
Ook zal er gekeken worden naar de voordelen van gecentraliseerde truststorebeheer en de eisen binnen de financiële sector.
Op het einde van de eerste fase zal er samen gezeten worden met de co-promotor om mogelijke oplossingen te bespreken die verder zullen worden uitgewerkt in de praktijkstudie.
De info die verkregen wordt in deze literatuurstudie zal de basis leggen voor de volgende fase van dit onderzoek: de praktijkstudie.
Deze fase zou starten in maart en loopt tot eind april met een duur van 8 weken.
Deze praktijkstudie opgedeeld in 3 delen.
het ontwerpen van een virtuele omgeving, de implementatie van het centraal truststorebeheer en de evaluatie van de oplossing.
In de eerste stap zal er weer worden samengezeten met de co-promotor om te bepalen welke componenten van het banknetwerk gesimuleerd moeten worden in de virtuele testomgeving.
Hierbij wordt gekeken naar de typische netwerkarchitectuur van een bank, inclusief servers (zoals applicatieservers en webservers), firewalls, proxyservers en verschillende besturingssystemen (zoals Linux en Windows).
Ook wordt er een certificate authority (CA) opgezet om het beheer van certificaten te simuleren.
Deze virtuele omgeving zal worden opgezet binnen GNS3, een netwerkemulator die het mogelijk maakt om complexe netwerken te simuleren.
Het doel is om een realistisch netwerk te creëren dat de complexiteit van een bankomgeving weerspiegelt.
De tweede stap, de implementatie van centraal truststorebeheer, begint midden maart en duurt drie weken.
In deze fase wordt een gecentraliseerde oplossing voor truststorebeheer opgezet.

Dit gebeurt door middel van de tools die gekozen werden samen met de co-promotor in de literatuurstudie in combinatie met zelfgeschreven scripts voor het certificaatbeheer.
Het doel hier is dat bij het genereren of updaten van een nieuw root certificaat, de truststores op elk systeem in het netwerk consistent zijn. Ook zullen er veiligheidscontroles geïmplementeerd worden, zoals het monitoren voor ongeldige of verlopen certificaten.
Na de implementatie volgt de derde stap van de praktijkstudie, die bestaat uit de evaluatie en validatie van de oplossing.
Deze fase begint begin april en duurt drie weken.
De centrale vraag is hoever de geïmplementeerde oplossing voldoet aan de beveiligingseisen (zoals DORA, PSD2, GDPR, ISO,...) van een bankennetwerk.
Er worden testcases opgesteld om veelvoorkomende uitdagingen te simuleren, zoals het herroepen van een rootcertificaat of het omgaan met verlopen certificaten.
Ook wordt de schaalbaarheid en efficiëntie van de oplossing getest.
Bij de schaalbaarheid zal er gekeken worden naar hoe de oplossing omgaat met het veranderen of uitbreiden van het netwerk. Bij de efficiëntie zal er gekeken worden naar hoe snel aanpassingen van certificaten binnen de centrale truststore verspreid worden binnen het netwerk zonder het verstoren van andere diensten.
Als laatste fase van dit onderzoek zal er een rapportage gemaakt worden. Dit zal gebeuren in het begin van mei met een tijdsduur van 4 weken. De rapportage zal de ontwerpkeuzes en oplossing beschrijven alsook de evaluatie en resultaten van deze oplossing.
Op basis van de bevindingen worden dan conclusies en aanbevelingen geformuleerd voor de implementatie van centraal truststorebeheer in banknetwerken.

A.4. Verwacht resultaat, conclusie

Dit onderzoek verwacht dat gecentraliseerd truststorebeheer verbeteringen oplevert in beveiliging, efficiëntie en schaalbaarheid voor een banknetwerk.
Door de certificaten op een centrale plaats te beheren, wordt het risico op verlopen of ongeldige certificaten verkleind, wat de volledige netwerkbeveiliging versterkt.
Het gecentraliseerde beheer zal ook de efficiëntie verhogen door automatisering van certificaatdistributie, wat tijd bespaart en menselijke fouten vermindert.
Bovendien maakt de oplossing schaalbaarheid mogelijk, wat essentieel is voor de groei van banknetwerken.
Dit onderzoek zal praktische aanbevelingen bieden voor de implementatie van deze oplossing in bankomgevingen, waardoor banken hun certificaatbeheer kunnen optimaliseren en hun netwerkbeveiliging kunnen versterken.

In conclusie verwacht dit onderzoek te bevestigen dat gecentraliseerd truststore-beheer niet alleen de veiligheid verbetert, maar ook bijdraagt aan een efficiënter en beter beheersbaar netwerk, met duidelijke voordelen voor de banksector.

Bibliografie

Arampatzis, A. (2020). What Is a Trust Store and How Hard Is It to Manage? *TLS certificates*. https://venafi.com/blog/what-trust-store-and-how-hard-it-manage/

Canonical. (2025). Install a root CA certificate in the trust store. *Ubuntu Server documentation*. Verkregen maart 6, 2025, van https://documentation.ubuntu.com/server/how-to/security/install-a-root-ca-certificate-in-the-trust-store/index.html

Digicert. (2025). What is a digitalcertificate? *Trust PKI*. https://www.digicert.com/faq/trust-and-pki/what-is-a-digital-certificate-and-why-are-digital-certificates-important

Dwivedi, D. (2024). What is a Trust Store and the Issues Associated with It. *Encryption Consulting*. https://www.encryptionconsulting.com/what-is-a-trust-store-and-the-issues-associated-with-it/

EncryptionConsulting. (2025). What is Certificate Enrollment and how is it used? Verkregen februari 26, 2025, van https://www.encryptionconsulting.com/education-center/what-is-certificate-enrollment-and-how-is-it-used/

E-Soft Inc. (2025). Mail (MX) Server Survey. *Research Reports*. Verkregen maart 12, 2025, van https://www.securityspace.com/s_survey/data/man.202502/mxsurvey.html

Gorobeț, I. (2024). Convergence of banking cybersecurity strategies to the new rules on digital operational resilience. *Proceedings of International Conference "Economic Security in the Context of Systemic Transformations"*. https://doi.org/10.53486/escst2023.06

Gounari, M., Stergiopoulos, G., Pipyros, K., & Gritzalis, D. (2024). Harmonizing open banking in the European Union: an analysis of PSD2 compliance and interrelation with cybersecurity frameworks and standards. *International Cybersecurity Law Review*, *5*(1), 79–120. https://doi.org/10.1365/s43439-023-00108-8

IBM. (2023, oktober). IBM z/OS Connect. https://www.ibm.com/docs/en/zos-connect/zosconnect/3.0?topic=connect-keystores-truststores

Internet Systems Consortium, Inc. (2025). Why use BIND 9? *BIND 9*. Verkregen maart 12, 2025, van https://www.isc.org/bind/

Mervana, P. (2024). What is a Trust Store and How to Manage It? *SSL Insight*. https://sslinsights.com/what-is-trust-store-and-how-to-manage-it/

Microsoft. (2024a). Trusted Root Certification Authorities Certificate Store. *Microsoft Learn*. Verkregen maart 6, 2025, van https://learn.microsoft.com/en-us/windows-hardware/drivers/install/trusted-root-certification-authorities-certificate-store

Microsoft. (2024b). What is Configuration Manager? *Microsoft Learn*. Verkregen maart 12, 2025, van https://learn.microsoft.com/en-us/mem/configmgr/core/understand/introduction

Microsoft. (2025). certutil. *Microsoft Learn*. Verkregen maart 6, 2025, van https://learn.microsoft.com/en-us/windows-server/administration/windows-commands/certutil

Mozilla. (z.d.). Network Security Services (NSS). *Firefox Source Docs*. Verkregen maart 6, 2025, van https://firefox-source-docs.mozilla.org/security/nss/index.html

Mozilla. (2018). NSS Shared DB. *Mozilla wiki*. Verkregen maart 6, 2025, van https://wiki.mozilla.org/NSS_Shared_DB

Mozilla. (2024). Set up Certificate Authorities (CAs) in Firefox. *Support Mozilla*. Verkregen maart 6, 2025, van https://support.mozilla.org/en-US/kb/setting-certificate-authorities-firefox

Netcraft. (2025). Web Server Survey. *January 2025 Web Server Survey*. Verkregen maart 12, 2025, van https://www.netcraft.com/blog/january-2025-web-server-survey/

Okta. (2023). What Is Certificate Verification for Root CA, Intermediate CA, and End-Entity CA. *Knowledge base*. Verkregen februari 26, 2025, van https://support.okta.com/help/s/article/How-can-we-do-Certificate-Verification-Need-more-explanation-for-Root-CA-Intermediate-CA-End-entity-CA?language=en_US

Perl, H., Fahl, S., & Smith, M. (2014). You Won't Be Needing These Any More: On Removing Unused Certificates from Trust Stores (N. Christin & R. Safavi-Naini, Red.). *Financial Cryptography and Data Security*, 307–315. https://doi.org/https://doi.org/10.1007/978-3-662-45472-5_20

Progress Software Corporation. (2024). Platform Overview. *Progress Chef Docs*. Verkregen maart 7, 2025, van https://docs.chef.io/platform_overview/

Red Hat. (2024a). 4.14. Using Shared System Certificates. *Red Hat Documentation*. Verkregen maart 6, 2025, van https://www.redhat.com/en/blog/configure-ca-trust-list

Red Hat. (2024b). Chapter 2. Introduction to Puppet. *OpenShift Container Platform*. Verkregen maart 6, 2025, van https://docs.redhat.com/en/documentation/openshift_container_platform/2/html/puppet_deployment_guide/chap-Introduction_to_Puppet#chap-Introduction_to_Puppet

Red Hat. (2025a). How Ansible works. *Ansible*. Verkregen maart 6, 2025, van https://www.redhat.com/en/ansible-collaborative/how-ansible-works

Red Hat. (2025b). Ansible Collaborative. *Ansible*. Verkregen maart 6, 2025, van https://www.redhat.com/en/ansible-collaborative

Reddy, R., & Wallace, C. (2010, oktober 1). Trust Anchor Management Requirements. https://doi.org/10.17487/RFC6024

SSL.com. (2024). What is the Role of a Certificate Authority? *What is a Certificate Authority (CA)?* Verkregen februari 25, 2025, van https://www.ssl.com/article/what-is-a-certificate-authority-ca/

Stack Exchange Inc. (2024). Technology. *2024 Developer Survey*. Verkregen maart 12, 2025, van https://survey.stackoverflow.co/2024/technology

Thales. (2025). What is PKI? *What is PKI and What is it used for?* Verkregen februari 26, 2025, van https://cpl.thalesgroup.com/faq/public-key-infrastructure-pki/what-public-key-infrastructure-pki

Topping, S. (2023). Understanding Public Key Infrastructure: Overview and Key Concepts. *GlobalSign Blog*. https://www.globalsign.com/en/blog/understanding-pki-overview-and-key-concepts

VMware. (2025). about Salt. *about Salt Project*. Verkregen maart 6, 2025, van https://docs.saltproject.io/en/latest/topics/about_salt_project.html#about-salt