Samenvatting theorie Cybersecurity Advanced

Prerequisites

1. OSI Model

De 7 lagen (van 1 → 7)

1. Physical – bits, kabels, elektrische signalen
2. Data Link – frames, MAC-adressen, switches
3. Network – IP-adressen, routing
4. Transport – TCP/UDP, segmenten
5. Session – verbindingen beheren
6. Presentation – encryptie, encoding
7. Application – HTTP, DNS, apps

Waar hoort wat?

Technologie	OSI-laag
MAC Address	Layer 2 – Data Link
IP Address	Layer 3 – Network
ARP	Geen laag
TCP	Layer 4 – Transport
UDP	Layer 4 – Transport
HTTP	Layer 7 – Application
ICMP (ping)	Layer 3 – Network

2. Routing en switching

Statische en dynamische IP-configuratie

Linux (CLI)

IP-adres instellen (tijdelijk):

```
1. sudo ip addr add 192.168.1.10/24 dev eth0
2. sudo ip route add default via 192.168.1.1
```

IP-adres via DHCP opvragen:

```
1. sudo dhclient eth0
```

IP-config bekijken:

```
1. ip addr
2. ip route
3. cat /etc/resolv.conf
```

Windows (CLI)

Statische IP instellen:

```
1. netsh interface ip set address "Ethernet" static 192.168.1.10 255.255.255.0 192.168.1.1
```

DNS instellen:

netsh interface ip set dns "Ethernet" static 1.1.1.1

DHCP inschakelen:

```
1. netsh interface ip set address "Ethernet" dhcp
2. netsh interface ip set dns "Ethernet" dhcp
3.
```

IP-config bekijken:

```
1. ipconfig /all
```

Default gateway

Moet de default gateway in je subnet zitten?

Ja, want anders is het niet mogelijk om de router te bereiken. De default gateway moet binnen hetzelfde subnet vallen als je host. Je kan de default gateway opvragen met de volgende commando's:

Windows: ipconfig

Linux: ip route (zoeken naar default via …)

DNS configuratie

Moet een DNS server in je subnet zitten?

Neen, dit is totaal niet nodig. De DNS mag extern zijn zoals de 8.8.8.8 van google of de 1.1.1.1 van cloudflare. Het enige wat belangrijk is, is dat deze server bereikbaar is.

Subnetting

Weten wat het netwerkdeel en het hostdeel is, om te kunnen zeggen of twee apparaten al dan niet in hetzelfde subnet liggen.

Snelle reminder:

- /24 = 256 adressen (254 bruikbaar)

- /25 = 128 adressen
- /30 = 4 adressen
- Netwerkadres = eerste
- Broadcast = laatste

3. Virtualbox netwerktypes

Type	Wat doet het?	Gebruik
NAT	VM kan naar internet, externen zien VM niet	Basic usage
Bridged	VM zit echt in je LAN, krijgt LAN-IP	Netwerk testen
Host-Only	Alleen host VM communicatie	Veilige labopstelling
Internal	Enkel tussen VMs	Geïsoleerde cyberlabs

4. Remote management

SSH verbinden

```
1. ssh user@ip
```

Belangrijke bestanden:

- ~/.ssh/authorized_keys → publieke keys die mogen inloggen
- ~/.ssh/known_hosts → fingerprints van hosts waarmee je al verbonden bent

5. Remote management

Symmetrische vs asymmetrische encryptie

Symmetrisch

- 1 sleutel voor encryptie & decryptie
- Snel
- Probleem: sleutel moet veilig gedeeld worden

Asymmetrisch

- Publieke sleutel → iedereen mag hebben
- Private sleutel → geheim houden
- Trager maar veilig voor key exchange

Public / Private key gebruik

- Public key → mag verspreid worden (bv. op een server in *authorized_keys*)
- Private key → blijft ALTIJD lokaal en geheim

CIA-triangle

Letter	Betekenis	Uitleg
Confidentiality	Vertrouwelijkheid	Enkel wie toegang mag hebben
Integrity	Integriteit	Data mag niet ongemerkt aangepast worden
Availability	Beschikbaarheid	Systemen moeten bereikbaar zijn

Network Address Translation:

- Zet private IP's om naar één public IP
- Laat meerdere toestellen één internetverbinding delen
- Types: Static, Dynamic, PAT (meest gebruikt)

Firewall

- Regelt wat binnen en buiten mag
- Werkt met regels (allow/deny)
- Types rules: IP, port, protocol, direction

Hoofdstuk I: DNS

Swiss Cheese model

Het Swiss Cheese-model stelt dat beveiliging bestaat uit **meerdere lagen,** zoals plakken kaas met gaten erin. Elke laag heeft kwetsbaarheden (gaten), maar pas wanneer de gaten op één lijn staan, geraakt een aanval helemaal door alle lagen heen.

Toepassing in Cybersecurity

- Meerdere beveiligingslagen verkleinen het risico op succesvolle aanvallen.
- Voorbeelden van lagen: firewalls, IDS/IPS, MFA, netwerksegmentatie, patching, least privilege, logging/monitoring.

- Een fout in één laag (bijv. slechte firewallregel) hoeft geen ramp te zijn als de andere lagen nog werken.

Types netwerkaanvallen

- DDoS / DoS
- DNS attacks (spoofing, poisoning, amplification)
- MITM (Man-in-the-Middle)
- Port scanning / reconnaissance
- ARP spoofing
- IP spoofing
- Bruteforce / credential stuffing
- Injection attacks (SQLi, command injection)
- Sniffing / packet capture
- Session hijacking
- Ransomware verspreiding via netwerk

Wat is een DDoS aanval?

DoS: Denial of Service → één bron overspoelt een server zodat hij niet meer kan reageren.
DDoS: Distributed Denial of Service → duizenden/miljoenen bronnen tegelijk.

Doelen:

- Service onbereikbaar maken
- Netwerk verzadigen (bandwidth flood)
- Server resources uitputten (CPU, RAM, sockets)

DNS als aanvalsvector

DNS kan misbruikt worden voor o.a.:

- DNS Spoofing/Poisoning → verkeerde IP's teruggeven
- DNS Amplification → DDoS versterkt via open resolvers
- DNS Tunneling → data exfiltratie door DNS-queries
- Zone Transfer → volledige domeinstructuur uitlekken
- Misconfigured DNS records → informatielek

DNS, theorie en tools

Uit een DNS server kan allerlei gevoelige informatie gehaald worden, voorbeelden hiervan zijn:

- IP-adressen (A/AAAA-records)
- Mailservers (MX)
- Nameservers (NS)
- Subdomeinen
- TXT-records (vaak gevoelige info: SPF, DKIM, API sleutels…)
- SOA-record met beheerinformatie
- Volledige zone via AXFR indien slecht geconfigureerd

Een normale DNS-resolve gaat als volgt. Een client vraagt aan de DNS server wat is het IP van hostname? Dit is intended behavior.
CLI tools om te praten met een DNS server zijn:

```
1. dig google.com
2. nslookup google.com
3.
```

Een reverse lookup doet dan weer het omgekeerde, deze vraagt het domein dat hoort bij een domein. Dit is enkel mogelijk als er PTR records bestaan in de DNS server.

Authoritative nameservers

- De primaire bron van waarheid voor een domein.
- Zij bevatten de DNS-zonefile.
- Non-authoritative antwoord = antwoord via caching resolver.

Zone transfer attack

Bij een zone transfer-aanval probeert een aanvaller een kopie te verkrijgen van het volledige DNS-zonebestand, dat informatie bevat over alle domeinnamen en subdomeinen in een bepaalde zone. Het wordt een aanval genoemd omdat de overdracht vaak ongeautoriseerd is. Zoneoverdrachten kunnen nuttig zijn voor aanvallers om een netwerk in kaart te brengen, maar niet alle zoneoverdrachten zijn schadelijk. Ze zijn bedoeld om DNS-records over meerdere gezaghebbende naamservers te verspreiden, maar als ze niet goed beveiligd zijn (bijvoorbeeld beperkt door IP), kunnen ze gevoelige informatie onthullen.

Tcpdump

Een netwerkdump maken kan met het commando:

```
1.  sudo tcpdump -i eth0 -w capture.pcap
```

Een bepaald protocol weglaten, in dit voorbeeld SSH:

```
1. sudo tcpdump -i eth0 not port 22 -w no_ssh.pcap
2.
```

Enkel HTTP verkeer opnemen

```
1. sudo tcpdump -i eth0 port 80 -w http.pcap
2.
```

Wireshark analyse

lle tcpdump filters werken ook in Wireshark.

Wat leert een analist uit Wireshark?

5.1 Conversations

- Laat zien welke hosts met elkaar communiceren
- Per protocol, IP of MAC
- Geeft inzicht in:
    - busiest talkers
    - mogelijke malware communicatie
    - volumepieken
    - verbindingen die niet thuishoren (C2 traffic)

5.2 Statistics

Bevat o.a.:

- Flow graphs (TCP handshake bekijken)
- I/O graphs (verkeer per seconde)
- Endpoints
- Packet lengths statistics

Hoofdstuk II: Firewalls

Firewalls in het OSI-model

Een traditionele (stateful) firewall werkt vooral op:

Layer 3 – Network

- Filtert op IP-adressen (bron/doel)
- Filtert op ICMP
- Routingbeslissingen

Layer 4 – Transport

- Filtert op TCP/UDP-poorten
- Houdt “state” bij (established connections)

Wat doet een traditionele firewall *niet*?

- Inspecteert geen payload of applicatiedata → dat is voor een NGFW (Next-Gen Firewall).

Types firewalls

We maken een onderscheid tussen twee verschillende soorten firewalls, zijnde host based en network based firewalls.

Host based firewalls

Bijvoorbeeld: Windows Firewall, UFW op Linux.

Voordelen

- Per-host controle (granulair)
- Beschermt tegen aanvallen binnen het netwerk
- Goed voor laptops en servers afzonderlijk

Nadelen

- Moeilijk om overal consistent te beheren
- Performance-impact op elke machine
- Moeilijk overzicht op netwerkbrede regels

Netwerk based firewalls

Staat aan de rand van het netwerk of tussen zones (Fortigate, pfSense, Cisco ASA…).

Voordelen

- Centraal beheer
- Schaalbaar
- Beschermt alle systemen tegelijk
- Goed voor netwerksegmentatie

Nadelen

- Single point of failure
- Onduidelijk wat intern gebeurt indien geen interne segmentatie
- Host-level aanvallen worden soms niet gedetecteerd

Netwerksegmentatie en Firewall Zones

Dit is de praktijk waarbij een netwerk wordt opgedeeld in kleinere subnetwerken om de **beveiliging en prestaties** te verbeteren. Door kritieke systemen te isoleren, beperk je de verspreiding van malware en verbeter je het beheer van netwerkverkeer.

Netwerk-/Firewallzones:
Zones zijn logische groepen van apparaten of netwerksegmenten die gelijke beveiligingsvereisten hebben.

DMZ: Demilitarized Zone

Een DMZ is een netwerkzone die **onbetrouwbare externe netwerken** (zoals het internet) scheidt van **vertrouwde interne netwerken**. In deze zone plaats je systemen die van buitenaf bereikbaar moeten zijn, zoals webservers of mailservers. Zo worden deze publieke systemen geïsoleerd van je interne netwerk.

DMZ met 1 firewall

Bij een opstelling met één firewall wordt de firewall uitgerust met **drie aparte interfaces/zones**:

- **WAN** – verbinding met het internet
- **DMZ** – publieke servers
- **LAN** – intern netwerk

De firewall bepaalt welke trafiek is toegestaan tussen deze zones (bijv. internet → webserver in DMZ, maar geen directe toegang tot het LAN).

NMAP

Statussen van de poorten:

Staat	Betekenis	Hoe nmap dit weet
Open	Service luistert	Krijgt SYN/ACK terug (bij TCP) of UDP response
Closed	Poort niet open	Krijgt RST (TCP reset) of ICMP Port Unreachable
Filtered	Firewall blokkeert	Krijgt niets terug of ICMP administratively prohibited

Banner grabbing is het achterhalen van informatie over de service die draait achter een bepaalde poort, zoals de versie of configuratie, dit kan worden gedaan met nmap.

Systemd

Configuration Files: Systemd configuration files are typically located in:

- /etc/systemd/system/ (for custom configurations).
- /lib/systemd/system/ (default configurations).

Systemctl Subcommands:

Commando	Doel
systemctl cat <service>	Toont configuratie incl. overrides
systemctl show <service>	Detail-info (properties)
systemctl edit <service>	Override file maken/bewerken
systemctl list-units	Actieve units
systemctl list-unit-files	Alle units op systeem
systemctl daemon-reload	Config opnieuw inladen nadat files zijn gewijzigd

Systemd times, dit zijn de vervangers van cron jobs. Ze bestaan uit 2 delen, eerst een timer en daarnaast een service die uitgevoerd wordt.

Proxies

Een **forward proxy** bevindt zich tussen de client en het internet. Alle verbindingen van de client verlopen via deze proxy, die functies kan uitvoeren zoals filtering, caching, anonimisering en toegangscontrole. Veelgebruikte voorbeelden zijn Squid en mitmproxy.

Een **reverse proxy** staat tussen het internet en interne servers. Externe bezoekers communiceren dus enkel met de reverse proxy, die de achterliggende servers verbergt. Reverse proxies worden gebruikt voor load balancing, SSL-termination, beveiligingsfilters en het afschermen van interne netwerkinformatie. Bekende software hiervoor is **Nginx**, **Apache**, **Traefik** en **HAProxy**.

Een **reverse proxy** kan vaak ook als load balancer functioneren. Waar een load balancer specifiek verkeer verdeelt over meerdere servers en health checks uitvoert, focust een reverse proxy op het routeren van verkeer naar interne systemen, caching, SSL-verwerking en het verbergen van backendservers. Tools zoals Nginx, HAProxy, Traefik en Envoy Proxy kunnen beide rollen combineren.

Hoofdstuk III: SSH

Correcte SSH configuratie draait rond het veilig en efficiënt verbinden met een remote machine. De belangrijkste elementen zijn:

1. Inloggen met keys; je genereert een private sleutel en een publieke sleutel. Je plaatst de publieke sleutel op de server. Hierdoor kan je inloggen zonder wachtwoord wat sneller en veiliger is.
2. Bestanden: authorized_keys en known_hosts
   - authorized_keys (op de server): bepaalt welke publieke sleutels toegang krijgen.
   - known_hosts (op de client): bevat fingerprints van servers waarmee je eerder verbinding maakte.
     → Dit voorkomt man-in-the-middle aanvallen.
3. SSH-configuratie voor client en server SSH-configuratie voor client en server
   Met een clientconfiguratie (~/.ssh/config) kan je shorthand-verbindingen maken, zoals: ssh web typen en automatisch inloggen.
4. **Passphrase vs wachtwoord**
   Een passphrase op een key beveiligt de *privékey* zelf.
   → Je hebt dus de key + passphrase nodig → extra beveiliging
   Een user/password authenticateert enkel de gebruiker, maar is minder veilig.

Jump / Bastion Host

Een jump host of bastion host is een speciaal, sterk beveiligd systeem dat dient als enige toegangspoort tot een intern netwerk.

Bedrijven gebruiken dit omdat:

- externe medewerkers niet rechtstreeks verbinding mogen maken met interne servers
- alle SSH-acties kunnen gelogd en gemonitord worden
- het aanvalsoppervlak beperkt blijft tot één gecontroleerde machine

Local vs remote port forwarding

Local port forwarden

Je maakt een lokale poort op je eigen machine die wordt doorgestuurd naar een service op een remote netwerk.

Use case:
Toegang krijgen tot een database die alleen intern bereikbaar is.

Je gebruikt lokaal poort **3307**, maar praat eigenlijk met de MySQL op de server.

Remote port forwarden

Je maakt een poort op de remote server die terug naar een service op je lokale machine verwijst.

Use case:

Je wil een demo-website op je laptop tijdelijk toegankelijk maken voor iemand anders via de server. Hier wordt poort 8080 op de server doorgestuurd naar je lokale webapp op poort 3000.

SOCKS protocol

Een SOCKS-proxy is een soort dynamische tunnel die al het netwerkverkeer via een SSH-verbinding routeert.

Gebruiksscenario's:

- Internetverkeer omleiden via een trusted netwerk (zoals een remote server)
- Veilig browsen op openbare wifi

- Toegang krijgen tot websites die enkel vanaf bepaalde locaties bereikbaar zijn

Het werkt als een algemene proxy voor je browser of applicaties.

Hoofdstuk IV: Honneypots

Wat is een honeypot?

Een **honeypot** is een systeem dat bewust kwetsbaar of interessant lijkt voor aanvallers, met als doel om hun gedrag te monitoren, aanvallen te detecteren of hen te vertragen. Het is geen productiesysteem, maar een lokmiddel.

Typen honeypots

1. Functioneel niveau

- **Low-interaction honeypots**
    - Simuleren slechts enkele protocollen of services.
    - Minder risico op compromittering.
    - Voorbeelden: Cowrie, Honeyd.
- **High-interaction honeypots**
    - Volwaardige systemen met echte services.
    - Realistisch gedrag → meer data, maar grotere risico's.
    - Voorbeeld: Een echte Windows/Linux VM met kwetsbaarheden.

2. Gebruik

- **Research honeypots**
    - Doel: Aanvallers begrijpen, nieuwe technieken ontdekken.
    - Worden vaak door universiteiten of securityteams gebruikt.
- **Production honeypots**
    - Doel: Dreigingen detecteren die echte systemen aanvallen.
    - Worden in bedrijfsnetwerken geplaatst.

3. Wat proberen ze te bereiken?

- Aanvallers misleiden.
- Aanvallen vroeg detecteren.

- Informatie verzamelen (TTP's, malware, technieken).
- Tijd winnen zodat detectie & verdediging kunnen reageren.

Honey/Canarytokens

Honeypot

- Volwaardig (of gesimuleerd) systeem.
- Aanvallers moeten er actief mee interageren.
- Richt zich op het lokken en observeren van aanvallers.

Honey-/Canarytoken

- Een **klein stukje data** dat alarm slaat wanneer het wordt gebruikt.
- Geen systeem, maar een *trigger*.
- Voorbeelden:
    - Unieke URL die een alert stuurt wanneer iemand klikt.
    - Valse API-key die alarm genereert wanneer iemand hem probeert te gebruiken.
    - File-tokens (bijv. "Passwords.xlsx") die waarschuwen wanneer geopend.

Verschil:
Honeypots lokken aanvallers naar een nep-systeem, tokens detecteren misbruik van *data*.

Docker

Is Docker virtualisation (type 1 or type 2), emulation, simulation?

Docker is geen virtualisatie, maar containerisatie. Het maakt gebruik van containers en is dus ook geen type 1 of type 2. Containers delen de host OS kernel maar isoleren applicaties en dependencies.

Containers delen de host-kernel → **lichte, snelle isolatie** maar minder strict dan VMs.

Is it considered to be more secure compared to virtual machines? Why (not)?

Nee, meestal niet.
Redenen:

- Containers delen dezelfde kernel → breuken kunnen de **host** compromitteren.
- Minder strikte isolatie dan een VM.
- Misconfiguratie (bijv. privileged containers) verhoogt risico's.

Voordelen t.o.v. VMs:

- Sneller.
- Lichtgewicht.
- Makkelijk herdeployable.

Maar qua beveiliging: VM > Container.

Is een honeypot in productie draaien met docker een slim idee?
Meestal geen goed idee, want:

- Als de honeypot wordt gehackt, kan de attacker *uit de container breken*.
- Containers zijn niet bedoeld als zware isolatielaag.
- In productie wil je geen verhoogd risico naar je host.

Wel bruikbaar voor labs, testen of research, maar niet voor een echte bedrijfsomgeving.

Docker socket in container: risico's

Als een container toegang heeft tot **/var/run/docker.sock**,
→ dan heeft hij **root-equivalent** toegang op de host.

Dat betekent:

- Containers starten/stoppen.
- Arbitrary code op de host uitvoeren.
- Privileged containers aanmaken.

Conclusie:
→ Dit geeft volledige controle over de host, en is **extreem onveilig** tenzij je de container vertrouwt.

Hoofdstuk V: Backups

Een goede back-upstrategie volgt meestal bepaalde principes of regels. De meest bekende is:

De 3-2-1 regel

- 3 kopieën van je data
- 2 verschillende media (cloud + HDD, of NAS + tape, …)
- 1 kopie off-site (andere locatie)

RPO & RTO

- RPO (Recovery Point Objective): hoeveel data mag maximaal verloren gaan? (bv. max. 1 uur)
- RTO (Recovery Time Objective): hoe snel moet je een systeem kunnen herstellen?

Back-up frequentie

- Dagelijks? Uurlijks? Continu?

Encryptie & toegang

- Back-ups moeten versleuteld zijn.
- Wie mag ze lezen/herstellen?

Testen van backups

- Een back-up die nooit getest wordt, is *geen* back-up.
- Regelmatig restore-tests uitvoeren.

What is the difference between a full vs an incremental backup? Give for both advantages and downsides.

Full Backup

Beschrijving:
Een volledige kopie van alle data.

Voordelen:

- Eenvoudig te restoren.
- Herstel is snel omdat alles in één bestand zit.

Nadelen:

- Duurt lang om te maken.
- Verbruikt veel opslagruimte.

Incremental Backup

Beschrijving:
Alleen de wijzigingen sinds de vorige backup (full *of* incremental).

Voordelen:

- Snelle back-ups.
- Minder opslagruimte nodig.

Nadelen:

- Herstel is trager: je hebt de full backup + ALLE incrementals nodig.
- Als één incremental beschadigd is → hele herstelketen kan falen.

Waarom is synchroniseren geen echte backup?
Cloudsync (OneDrive, Dropbox, Google Drive) is géén back-up, omdat:

- Synchronisatie = wijziging → overal gewijzigd
  → Als je een bestand wist of ransomware het versleutelt, synchroniseert de cloud exact hetzelfde (foute) bestand mee.
- Geen echte versiehistoriek (of te beperkt).
- Geen off-site *onafhankelijke* kopie.
- Geen bescherming tegen menselijke fouten.
- Niet ontworpen om *volledige systemen* te herstellen.

Kort: cloudsync = mirror, geen backup.

Waarom niet 100% vertrouwen op één cloudprovider?
Cybersecurity-experts waarschuwen dat je **nooit afhankelijk mag zijn van één opslagplaats**, zelfs niet “in de cloud”. Redenen:

Single point of failure

Als OneDrive crasht, gehackt wordt, of een fout maakt → al je data kwijt.

Ransomware kan ook je cloud treffen

- Versleutelde bestanden worden **gesynchroniseerd** naar de cloud.
- Ransomware kan **tokens of accounts misbruiken** om alles te wissen.

Menselijke fouten

- Per ongeluk delete = delete overal.
- Cloudproviders herstellen soms niet alles.

Accountverlies

- Account gehackt → aanvaller wist alles.
- Account geblokkeerd (betalingsfout, regio-issues).

Juridische of organisatorische risico's

- Het bedrijf kan failliet gaan of diensten stopzetten.
- Compliance-regels (GDPR) kunnen migratie vereisen.
- Provider bepaalt retentie, niet jij.

Geen volledige controle

Je weet niet:

- waar de data staat,
- hoe vaak het wordt geback-upt,
- hoe lang retentie is,
- wie toegang heeft.

Conclusie:
Cloud ≠ perfecte, betrouwbare back-up.
Altijd **meerdere kopieën, op verschillende plaatsen**, onder jouw eigen controle.

Hoofdstuk VI: CA-HTTPS

HTTPS-foutmelding: verkeer nog steeds versleuteld?

Ja.
Zelfs als je browser zegt dat een certificaat ongeldig is (bv. verlopen) en je toch doorgaat, dan:

- **De verbinding blijft versleuteld.**
- **Maar je kan de identiteit van de server niet vertrouwen.**

Risico: je *denkt* met de echte site te praten, maar iemand kan een Man-in-the-Middle zijn met een fake certificaat.

Encryptie ≠ Vertrouwde identiteit.

Wat is X.509?
X.509 is de internationale **standaard** die beschrijft hoe digitale certificaten moeten worden opgebouwd:

- Publieke sleutel
- Identiteit (CN, SAN, organisatie…)
- Handtekening van de CA
- Geldigheidsperiode
- Extensions (bv. Key Usage, SAN)

Alles wat je *in een certificaat* ziet, volgt X.509.

Wat is een CSR?
Certificate Signing Request:

- Een bestand dat je aanmaakt wanneer je een certificaat wil aanvragen.
- Het bevat:
    - je **publieke sleutel**
    - de **gegevens van je server** (CN, SAN,…)
- Je stuurt het naar een CA, die er een geldig certificaat van maakt.
- De **private key blijft altijd lokaal**.

Wat is SAN (Subject Alternative Name)?
Een X.509 extensie waarmee je **meerdere domeinen** in één certificaat kan opnemen.

Voorbeeld:

- example.com
- www.example.com
- api.example.com

Tegenwoordig belangrijker dan CN.

Certificaatketens & Cross-certification
Certificate Chain

De keten bestaat uit:

1. **Root CA** (vertrouwd in je OS/browser)
2. **Intermediate CA** (ondertekend door root)
3. **Servercertificaat**

Browsers vertrouwen je servercertificaat omdat er een **ononderbroken keten** is naar een vertrouwde root.

Cross-Certification (CA-vernieuwing)

Wanneer een root CA bijna verloopt, wordt:

- een **nieuw rootcertificaat** aangemaakt
- het nieuwe rootcertificaat wordt **ondertekend door het oude rootcertificaat**

Dit zorgt voor:

- een overgangsperiode waarin oude apparaten die de nieuwe root nog niet kennen toch nog vertrouwen hebben via het oude rootcertificaat.

Verschil tussen SSL en TLS

- **SSL** is de oude, verouderde en onveilige voorloper.
- **TLS** is de moderne standaard.

Huidige standaard: TLS 1.3

MD5?

Nee.
MD5 is *insecure* en wordt niet meer gebruikt in moderne TLS-versies.

Wat is Let's Encrypt?
Een gratis, geautomatiseerde Certificate Authority die:

- domeinvalidatie uitvoert (ACME-protocol)
- gratis, betrouwbare certificaten uitgeeft
- certificaten met korte geldigheidsduur (90 dagen) gebruikt voor veiligheid

Is dat niet gevaarlijk (hackers krijgen gratis certificaten)?

Nee, want:

- Een certificaat zegt *niets* over intenties.

- Het bevestigt alleen: **de persoon die het aanvraagt, controleert het domein.**
- Malware kan perfect ook op domeinen met betaalde certificaten draaien.
- Door goede encryptie wordt het internet veiliger voor iedereen, ook voor legitieme gebruikers.

Hoe kan je HTTPS-verkeer decrypten bij TLS 1.2?
Met de **private key** van de server kun je:

- TLS 1.2 sessies ontsleutelen
- zolang er **RSA key exchange** gebruikt wordt
  (server private key decrypt de session key)

Vb. in Wireshark:

- (Pre-)Master Secret Key → hiermee kan je alles decoderen.

Waarom werkt dit niet bij TLS 1.3?
TLS 1.3 gebruikt **altijd**:

- Elliptic Curve Diffie-Hellman (ECDHE)
- Perfect Forward Secrecy is verplicht

Daarom:

- De *private sleutel van de server* kan **niet** meer worden gebruikt om sessies te decoderen.
- De session keys worden per sessie gegenereerd en nooit gedeeld.

Gevolg:
Passive decryption zoals in TLS 1.2 is onmogelijk.

Kan een bedrijf TLS 1.3 toch nog volledig inspecteren?
Ja, maar **niet via passieve decryptie**.

Ze gebruiken:

SSL/TLS Interception (MITM-proxy)

Bedrijven plaatsen:

- een **enterprise root CA** op alle toestellen
- een **TLS-inspectieproxy** in het netwerk

Werking:

1. Gebruiker maakt HTTPS-verbinding → proxy onderschept.
2. Proxy maakt **nieuwe TLS-verbinding** naar de echte server.
3. Proxy decrypt → scant → re-encrypt.
4. Geen trustfouten omdat de proxy een trusted enterprise CA gebruikt.

Zo kan een bedrijf **alle TLS-versies, inclusief TLS 1.3**, inspecteren.

Hoofdstuk VII: SIEM

Wat is een SIEM?

- Acronym**:** Security Information and Event Management.
- Doel**:** Het centraal verzamelen, correleren, analyseren en visualiseren van logs en beveiligingsgebeurtenissen.
- Functionaliteiten**:** Logverwerking, correlatie-regels, detectie van afwijkend gedrag, dashboards, alerts.
- Voorbeelden/Tools:
    - Wazuh, Splunk, Elastic SIEM, IBM QRadar, Azure Sentinel (Defender for SIEM).

Wat is een SOC?

Acronym: *Security Operations Center*.

Doel: Afdeling/team dat verantwoordelijk is voor het **monitoren**, **detecteren**, **analyseren**, **beantwoorden** en **beheren** van beveiligingsincidenten.

Bestaat uit**:** Analisten (Tier 1–3), incident responders, threat hunters.

Wat betekent compliancy in cybersecurity?

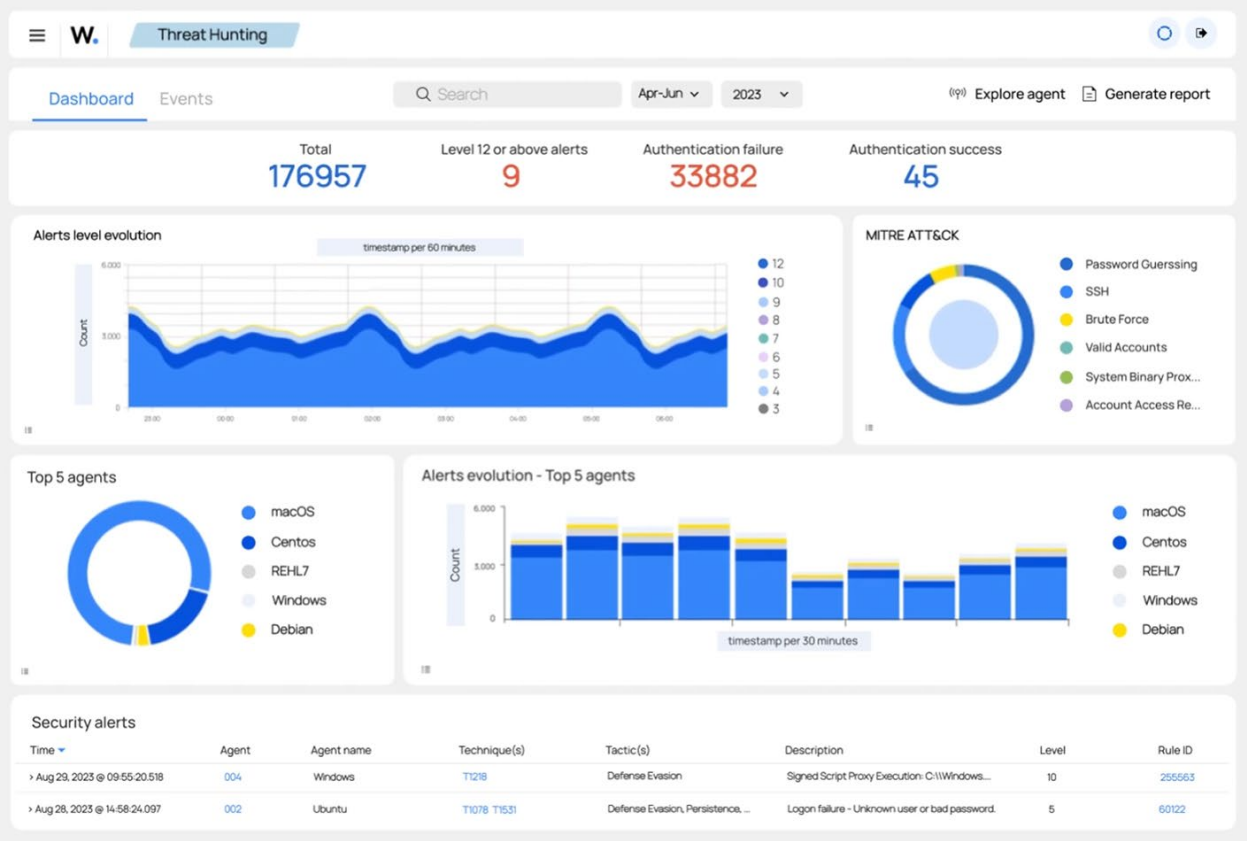
- Voldoen aan **wetten**, **reguleringen**, **standaarden** en **policies** rond beveiliging.
- Voorbeelden: **GDPR**, **ISO 27001**, **NIST**, **PCI-DSS**.
- Het gaat om **aantoonbaar** voldoen aan vastgelegde security-vereisten.

Welke features biedt Wazuh?

Wazuh is een open-source SIEM/XDR-platform met:

- Logverzameling en analyse
- File Integrity Monitoring (FIM)

- Intrusion Detection (host-based IDS)
- Malware detection & rootkit detection
- Compliance checks (PCI, GDPR, CIS…)
- Vulnerability detection
- Configuration assessment
- SIEM dashboards via Kibana / OpenSearch
- Alerting & correlation rules



Wat is FIM?

FIM staat voor File Integrity Monitoring, het is een technologie die wijzigingen in bestanden kan dedecteren zowel: creatie, aanpassing als verwijdering. Belangrijk voor detectie van malware, rootkits, ongeautoriseerde configuratiewijzigingen.

Wat is Sysmon?

Het is een onderdeel van de Windows Sysinternals suite. Het logt gedetailleerde systeemevents zoals procescreatie, netwerkverbindingen,

bestandswijzigingen. Wordt gebruikt voor threat hunting en forensics. En stuurt logs naar Event Viewer → kan door een SIEM verwerkt worden.

Verschil tussen IDS en IPS

IDS	IPS
Intrusion Detection System	Intrusion Prevention System
Detecteert verdachte activiteit	Detecteert én blokkeert
Passief	Actief
Stuurt alerts	Kan verkeer droppen of aanpassen
Minder impact bij foutmelding	Kans op false positives → legitiem verkeer geblokkeerd

Fundamenteel verschil tussen firewall en IDS/IPS

Firewall

- Gericht op toegang controleren op basis van IP, poorten, protocollen.
- Werkt meestal op laag 3 & 4 (network/transport).
- Staat in voor permit/deny van verkeer.

IDS/IPS

- Gericht op detectie (en preventie bij IPS) van schadelijke inhoud.
- Analyseert pakketinhoud (deep packet inspection, laag 7).
- Kan specifieke aanvalspatronen herkennen (signatures, anomalies).

Basis Suricata-regels interpreteren

Detecteer specifiek protocol:

```
1. alert http any any -> any any (msg:"HTTP detected"; sid:100001;)
2.
```

Uitleg: Triggert wanneer HTTP-verkeer gedetecteerd wordt.

Detecteer verkeer op specifieke poort

```
1. alert tcp any any -> any 22 (msg:"SSH traffic detected"; sid:100002;)
2.
```

Uitleg: Alert wanneer verkeer naar poort 22 (SSH) gaat.

Waar plaats je een fysieke IDS/IPS in het netwerk?

IDS (passief):

- Plaats je **in een SPAN/mirror poort** op een switch zodat het **verkeer kan meelezen**.
- Geen impact op netwerkperformance (want passief).

IPS (inline):

- Plaats je **tussen firewall en interne switch**, zodat het **al het verkeer kan inspecteren en blokkeren**.
- Wordt een **single point of failure**.
- **Impact:**
    - Hogere latentie
    - Bij fout → netwerkverkeer kan stoppen
    - Vereist redundantie (cluster)

Wat is Security Onion?
Een open-source Linux-distributie gericht op threat hunting, monitoring en incident response.

- Combineert verschillende tools zoals:
    - Zeek (network analysis)
    - Suricata (IDS/IPS)
    - Elasticsearch/Kibana (SIEM dashboards)
    - Wazuh integratie
    - TheHive (incident management)

Doel van Security Onion

- Een all-in-one platform voor:
    - Netwerkmonitoring
    - Hostmonitoring
    - Intrusion detection
    - Logbeheer
    - Threat hunting
    - Incident response

Hoofdstuk VIII: IPsec

Wat is het fundamentele doel van IPsec?

Het doel van IPsec is om **netwerkverkeer op IP-niveau te beveiligen** door middel van:

- **Confidentiality** (versleuteling)
- **Integrity** (bescherming tegen manipulatie)
- **Authentication** (zeker weten wie de zender is)
- **Anti-replay protection**

IPsec wordt gebruikt voor **VPN's**, **site-to-site tunnels**, **remote access**, …

Hoe werkt IPsec? Wat zijn SP's en SA's?

Security Policy

- Beschrijft **welke pakketten** beveiligd moeten worden en **hoe**.
- Inhoud: bron/dest IP, protocol, poorten, en welke IPsec-acties toegepast worden (encryptie, authenticatie…).

Security Association

- Een **"contract" tussen twee hosts/gateways** dat alle **cryptografische parameters** bevat:
    - Encryptie-algoritme
    - Authenticatie-algoritme
    - Sleutels
    - SPI (Security Parameter Index)
- Een SA is **unidirectioneel** → je hebt er minstens **2** nodig voor bidirectioneel verkeer.

Hoe IPsec werkt in het kort:

1. **IKE** onderhandelt sleutels en maakt SA's aan.
2. IPsec gebruikt die SA's om verkeer te **authenticeren / versleutelen** volgens de SP's.
3. Het verkeer wordt beschermd met **AH of ESP**.

Welke modes bestaan in IPsec? Wat is het verschil?

Transport Mode

- Alleen de payload van het IP-pakket wordt beveiligd.
- Het originele IP-adres blijft zichtbaar.
- Gebruikt voor host-to-host communicatie.

Tunnel Mode

- Het volledige IP-pakket wordt ingekapseld in een nieuw IP-pakket.
- Origineel IP-adres is verborgen.
- Gebruikt tussen gateways/firewalls (site-to-site VPN).
- Veiliger voor netwerken.

Kort:

- Transport mode = payload beveiligen
- Tunnel mode = volledig pakket beveiligen

Wat is AH (authentication header)?
AH = Authenticatie en integriteitsbescherming.

Kenmerken:

- Geeft **integrity**, **authentication**, **anti-replay**.
- **Geen encryptie** → inhoud blijft zichtbaar.
- Beschermt ook delen van de **IP-header**.
- Protocolnummer: **51**

Wordt vandaag minder gebruikt omdat ESP "alles" kan.

Wat is ESP (Encapsulating Security Payload)?
ESP = Encryptie + optionele authenticatie.

Kenmerken:

- **Encryptie** (confidentiality)
- **Integrity & authentication** (optioneel, maar meestal aan)
- Verbergt payload volledig
- Protocolnummer: **50**

ESP is het **meest gebruikte IPsec-protocol**

Wat is IKE?
IKE = Internet Key Exchange.

Doel van IKE:

- **Onderhandelen van cryptografische parameters**
- **Uitwisselen van sleutels**
- **Opzetten van Security Associations (SA's)** voor IPsec

IKE werkt in **twee fasen**:

1. **IKE Phase 1**
    - Opzetten beveiligd kanaal (ISAKMP SA).
2. **IKE Phase 2**
    - Onderhandelen van IPsec SA's (voor AH/ESP).

Modern gebruik: **IKEv2** (sneller, stabieler, veiliger).

Hoofdstuk IX: VPN

Wat zijn enkele nadelen / beperkingen van IPsec?
Complexiteit: veel protocollen (IKE, AH, ESP), veel parameters (algoritmes, lifetimes, modes, policies). Moeilijker te configureren en debuggen.

Interoperabiliteit: verschillende implementaties/opties kunnen incompatibiliteit veroorzaken (crypto-sets, NAT-T, XAuth).

NAT-problemen: originele IPsec (ESP/AH) heeft problemen met NAT; NAT-T en UDP-encapsulation lossen veel gevallen op maar voegen complexiteit/overhead toe.

Key management overhead: IKE-configuraties, certificaatbeheer of pre-shared keys voor veel endpoints is werkintensief.

Performance / overhead: crypto + encapsulation = extra CPU en mogelijke MTU/fragmentatie-issues → throughput & latency impact.

Statefulness / troubleshooting: moeilijk te inspecteren/monitoren omdat verkeer versleuteld en ingekapseld is.

Deployment grenzen: host-to-host transport mode versus gateway tunnel mode vergt verschillende ontwerpen.

Wat is een CA en hoe werkt die?

CA (Certificate Authority) = entiteit die digitale certificaten uitgeeft en daarmee public keys bindt aan identiteiten (domeinnamen, organisaties, personen).

Belangrijkste onderdelen / flow:

1. **Keypair en CSR**: Een server (bv. chamilo.hogent.be) genereert een private key + public key en stuurt een CSR (Certificate Signing Request) naar de CA met de publieke sleutel en identificatiegegevens (CN / SAN).
2. **Verificatie**: CA verifieert identiteit (domain control, organisatie, afhankelijk van certificaattype).
3. **Ondertekenen**: CA maakt een certificaat (X.509) met publieke sleutel + identiteit en signeert het met de CA-private key.
4. **Certificaatketen**: vaak wordt er een keten gebruikt: end-entity cert ← intermediate CA ← root CA. Root CA is in trust stores (browsers/OS). Intermediates verminderen risico bij compromittering.
5. **Trust en validatie**: Client vertrouwt certificatetekens wanneer de keten eindigt in een root CA die in zijn truststore staat, de datum correct is, de naam matcht, en het cert niet is ingetrokken.
6. **Revocation**: CRL / OCSP (controleerbaarheidsmechanismen) of OCSP stapling door de server.
7. **Automatisering**: ACME (Let's Encrypt) automatiseert CSR/validation/renewal.

Waarom krijgen we geen browser-waarschuwing voor https://chamilo.hogent.be?

(Stap-voor-stap, client perspective)

1. **DNS-resolutie**: browser vraagt DNS → krijgt IP van chamilo.hogent.be.
2. **TCP connect**: browser opent TCP (of UDP/QUIC) naar poort 443 op die IP.
3. **TLS handshake begint**:
    - **ClientHello**: browser stuurt ondersteunde TLS-versies, cipher suites, en SNI (server name = chamilo.hogent.be).
    - **ServerHello**: server kiest TLS-versie/cipher suite.

    - **Server sends Certificate**: server stuurt zijn X.509-certificaat (en intermediate certs als nodig).
4. **Certificate validation (door browser/OS)**:
    - **Chain building**: browser bouwt keten naar een vertrouwde root CA (in truststore).
    - **Naamcontrole**: het certificaat bevat chamilo.hogent.be in CN of SAN; moet matchen op SNI/hostname.
    - **Datumcontrole**: certificaat is binnen geldigheidsperiode.
    - **Chain signatures**: elk certificaat in de keten is cryptografisch ondertekend door de volgende CA.
    - **Revocation check**: (optioneel / afhankelijk) browser controleert CRL/OCSP/OCSP-stapling. Als alles ok → certificate accepted.
5. **Key exchange & session keys**: client en server ronden key exchange (bijv. ECDHE) af nu worden symmetrische keys afgeleid.
6. **Finished messages**: beide kanten verifiëren handshake integrity.
7. **Encrypted HTTP(S) traffic**: browser en server wisselen versleutelde HTTP(S)-verzoeken/antwoorden.

Dus: geen waarschuwing omdat: het certificaat geldig is, de naam klopt, de keten eindigt in een vertrouwde root, en revocation checks (voor zover vereist) niet faalden.

Extra punten:

- **SNI** zorgt dat server meerdere certificaten voor verschillende hostnames kan presenteren op dezelfde IP.
- **HSTS** kan toekomstige man-in-the-middle waarschuwingen verminderen door HTTP→HTTPS afdwingen.
- **Automatische vernieuwing** (bv. Let's Encrypt + ACME) voorkomt verlopen certificaten.

Wat is het doel van OpenVPN?

- **Doel:** veilige, flexibele VPN-oplossing voor tunneling van netwerkverkeer over het internet, zodat hosts/netwerken veilig kunnen communiceren alsof ze op hetzelfde LAN zitten.

- Gebruik: remote access, site-to-site tunnels, toegang achter NAT/firewalls (met UDP/TCP)

Hoe werkt OpenVPN? Cruciale elementen voor een werkende OpenVPN setup

Hoog niveau werking

- OpenVPN creëert een **versleutelde tunnel** (via TLS for control channel + symmetric crypto for data) tussen client en server of tussen peers.
- Kan werken over UDP of TCP. Kan tun (layer 3) of tap (layer 2) interfaces creëren.

Cruciale elementen

1. **Tun/tap interface**: virtuele netwerkinterface op client & server. tun = routed IP layer (layer 3). tap = Ethernet bridging (layer 2).
2. **Authenticatie & encryptie**: meestal TLS met PKI (server cert + client certs) of met pre-shared keys. OpenVPN gebruikt TLS handshake (server cert) en vervolgens generieke symmetric ciphers for data.
3. **PKI/Certificates or PSK**: CA + server cert + client certs (meest gebruikt) of shared key mode.
4. **Server config**: IP pool for VPN clients, routing/pushing routes, server or server-bridge directive.
5. **Client config**: correcte server IP/name, certificates/keys, cipher settings, routes.
6. **Ports & firewall**: UDP/TCP port open on server (default UDP 1194) and NAT/firewall rules.
7. **Routing or bridging**: configure routes on server or push routes to clients. If you want access to a private LAN behind the server, configure IP forwarding and iptables/NAT as needed.
8. **Compression (optional)**: deprecated in many configs due to CRIME-like issues.
9. **Keepalive & TLS-auth**: tls-auth or tls-crypt HMAC for extra anti-replay and to protect the TLS handshake from unauthenticated packets (also helps with DDoS).

10. **Client/server key rotation & revocation**: use CRL to revoke compromised client certs.

Wat is PKI?

PKI (Public Key Infrastructure) = set van rollen, policies en procedures nodig om public/private keypairs te beheren en te vertrouwen:

- Componenten: CAs (root + intermediates), Registration Authorities (optioneel), certificaat database, revocation mechanisms (CRL/OCSP), certificate policies, lifecycle management (issuance, renewal, revocation).
- Doel: betrouwbaarheid van public keys garanderen en veilige sleutel-uitwisseling mogelijk maken

Fundamentele verschillen: IPsec vs OpenVPN

Werken ze op dezelfde OSI-layer?

- **IPsec**: werkt op **network layer (Layer 3)** — beschermt IP-pakketten; tunnel/transport modes.
- **OpenVPN**: typisch werkt op **user-space** en creëert **tun (L3) of tap (L2)**; functioneel levert het VPN op L3 of L2. Dus beide kunnen op L3 werken, maar OpenVPN is een applicatie in userland die virtual network interfaces gebruikt. IPsec werkt “onder” de applicaties op OS/kernel niveau (veelal kernel-implemented).

Hebben ze hetzelfde doel?

- **Overlappend doel**: beide bieden **veilige tunnels/VPN** over onveilige netwerken — confidentiality, integrity, authentication.
- **Verschil in scope**: IPsec is een gestandaardiseerde suite (breed inzetbaar op router/gateway niveau, ingebakken in OSes), ideaal voor **site-to-site** gateway-to-gateway. OpenVPN is flexibeler/eenvoudiger te gebruiken voor **remote access** en client-to-server scenarios, en is makkelijker door NAT/firewall heen (werkt over UDP/TCP).

Redenen om één boven de ander te gebruiken

Kies IPsec als:

- Je wil **gateway-to-gateway** (site-to-site) met native OS/widely supported protocol.

- Je wil gebruikmaken van ingebouwde OS ondersteuning op routers/firewalls.
- Je hebt performance eisen en kunt kernel-acceleratie van IPsec gebruiken (ESP in kernel).
  Nadelen IPsec: configuratie complexer, NAT-T issues, meer beleidswerk.

Kies OpenVPN als:

- Je wil eenvoudige client configuratie en makkelijk door NAT/firewall (TCP/UDP, configurable ports).
- Je wil L2 bridging (tap) of L3 routing vanuit userland.
- Je wilt flexibele auth (certificates, username/password, MFA).
  Nadelen OpenVPN: user-space overhead (iets minder performant dan goed geconfigureerd IPsec), extra software nodig op endpoints.

Is WireGuard meer vergelijkbaar met OpenVPN of IPsec of beide?

Architectuur & doel: WireGuard is een modern, lichtgewicht VPN-protocol dat **op Layer 3** werkt (zoals IPsec). Het vervangt traditioneel IPsec/OpenVPN in veel use-cases door eenvoud en performance.

Vergelijking met IPsec: functioneel meer vergelijkbaar — WireGuard werkt op netwerklaag (IP tunneling), maakt gebruik van kernel-implementatie in veel OSes (zeer snel), en is gericht op minimale attack surface.

Vergelijking met OpenVPN: qua gebruiksgemak en eenvoudige configuratie lijkt WireGuard op OpenVPN (beide eenvoudig te configureren in veel scenario's), maar WireGuard is **simpeler** en moderner, met een klein, auditbaar codebase. WireGuard gebruikt modern crypto (Noise protocol concepts), geen complexe IKE.

Belangrijk onderscheid:

- **WireGuard ≈ IPsec** in laag (L3) en prestatieniveau.
- **WireGuard ≈ OpenVPN** in gebruiksvriendelijkheid en eenvoud van deployment (maar zonder OpenVPN's TLS-based authentication architecture).

Concreet: WireGuard is het beste van beide werelden: eenvoud & performance. Het heeft geen (standaard) built-in PKI zoals OpenVPN/TLS (config is op publieke sleutels), en lacks sommige features van IPsec (complex

policy control, veel enterprise features) maar er bestaan aanvullende tools waaronder key management en integration.