



# Functioneel Ontwerp - Project Cyber Security

## Inleiding

### Waarom een SIEM?

Het midden- en kleinbedrijf (mkb) is kwetsbaar voor cyberaanvallen door beperkte middelen en kennis. Wetgeving zoals de NIS2-richtlijn verplicht organisaties tot netwerkmonitoring, logging en incidentrespons. Een SIEM-oplossing (Security Information and Event Management) centraliseert loggegevens en maakt snelle detectie van dreigingen mogelijk. Na onderzoek is gekozen voor Wazuh, een open-source SIEM-oplossing die goed aansluit bij de behoeften van het mkb.

### Doelgroep

IT-verantwoordelijken binnen het mkb, zoals systeem- en netwerkbeheerders. Ook geschikt als referentie voor externe consultants.

### Uitgangspunten

Er is uitgegaan van een gemiddelde mkb-omgeving met beperkte IT-infrastructuur en een klein beheerteam. De oplossing moet laagdrempelig, schaalbaar en wettelijk compliant zijn.

## Architectuur van de SIEM-oplossing

De architectuur omvat logbronnen, agents, een centrale manager en een dashboard. Loggegevens worden verzameld, gecorreleerd en geanalyseerd. Alerts worden gegenereerd bij afwijkend gedrag volgens ingestelde detectieregels.

Link naar architectuurdiagram: [Mermaid Chart](#)

## Bedrijfsregels en Validaties

## Bedrijfsregels

1. Minimaal 3 typen logbronnen gekoppeld.
2. Per logbron minimaal 5 detectieregels.
3. Alleen geautoriseerde gebruikers mogen detectieregels aanpassen.
4. Alerts alleen bij afwijkingen binnen parameters.
5. Elke alert moet automatisch gelogd worden.

## Detectievereisten

1. Minimaal 5 detectieregels per logbron.
2. Detectieregels bevatten prioriteitstoekenning.
3. Gebruik van gedrags-, tijd- of patroonanalyse.
4. Gebruik van gestandaardiseerde regels waar mogelijk.
5. Detectieregels moeten versiebeheer ondersteunen.

## Validaties

- Tijdsvalidatie
- Categoriecontrole
- IP/locatiecheck
- Gebruikersrechtencheck
- Structuurvalidatie

## Ondersteunende functies

- Zoekfunctionaliteit
- Versiebeheer van detectieregels
- Notificatiesysteem
- Prioritering en filtering

## Detectiescenario's per Logbron

### Authenticatie- en toegangslogs

- Toevoeging aan administratorsgroep

*Voorbeeld over de uitwerking van een detectiescenario. Het volledige Functioneel Ontwerp is op te vragen bij de projectgroep*

## **Scenario: Toevoeging aan administratorsgroep**

Dit scenario richt zich op het detecteren van het escaleren van beheerdersrechten, waarbij een gebruiker wordt toegevoegd aan de lokale administratorsgroep op een Windows systeem. Deze actie geeft een aanvallers volledig controle over het systeem en is een veelgebruikte methode voor escalatie na toegang tot het systeem.

### **Doel van de detectie**

Het detecteren van pogingen tot escalatie door het toevoegen van een gebruiker aan de lokale administratorsgroep op een Windows 11 systeem.

### **Detectiecriteria**

Een alert moet afgaan als:

- Een gebruiker wordt toegevoegd aan de groep "Administrators"

### **Voorwaarden voor de detectie**

- User Accountmanagement auditing moet geactiveerd zijn op het systeem
- De Wazuh-agent moet het event verzamelen
- De Wazuh-regel moet geconfigureerd en geactiveerd zijn

### **Databronnen**

- Windows Event logs: Registreren van wijzigingen aan groepen
- Wazuh-agent: Verzamelt deze events en stuurt dit naar de Wazuh-manager
- Wazuh-manager: Ontvangt de logs en triggert detectieregels

### **Verwacht gedrag**

Wanneer een gebruiker wordt toegevoegd aan de lokale administratorsgroep registreert Windows dit in de Windows Eventlogs. De Wazuh-agent leest dit in de logs en stuurt dit door naar de Wazuh-Manager. De Manager vergelijkt dit met detectieregels en als het gedrag overeenkomt wordt er een alert gegenereerd en getoond op het Wazuh-dashboard.

*Verdere scenario's uit deze logbrongrop:*

- Password spraying
- Credential attacks
- Bruteforce via RDP
- Wissen van Windows Security Logs

## Endpointbeveiligingslogs (AV/EDR)

- Detectie van malware
- Uitschakelen van beveiliging
- Persistente aanvallen via Registry Runkeys
- PowerShell-gebruik
- Timestomping
- Verdachte scripts in /tmp of gebruikersfolders

## Firewall- en netwerklogs

- Uitschakelen Windows Firewall
- Misbruik van Windows Task Scheduler
- Inlogpoging buiten werktijden
- Inlogpoging vanaf vreemd netwerk
- AD-informatie ophalen via PowerShell

## Rollen en Verantwoordelijkheden

Rol	Taken
SIEM-beheerder	Installatie, configuratie, logbronbeheer, detectieregels beheren
Security Analyst	Monitoring, analyse, opvolging incidenten
Systeembeheerder	Agentinstallatie, infrastructuurbeheer
Functioneel Beheerder	Afstemming op informatiebehoeften, compliancy

## RACI-matrix

Zie oorspronkelijke document voor uitgebreide tabel. Rollen zijn niet strikt gescheiden, aanpasbaar op organisatiegrootte.

# Autorisatiematrix

Rol	C	R	U	D
Functioneel beheerder		X		
Systeembeheerder	X	X	X	
Security analist		X	(optioneel)	
SIEM-beheerder	X	X	X	X

# Interfaces

## Componenten

- Wazuh Server, Dashboard, Indexer, Agents
- Netwerkkapparatuur, authenticatiesystemen, endpoints

## Koppelingen

- Netwerkkapparatuur → Wazuh Server
- Endpoints → Wazuh Agent → Wazuh Server

## Toegang beheerder

Toegang tot Wazuh-dashboard via browser, waar logs en meldingen visueel worden weergegeven.


# Samenvatting en Aanbevelingen

- **Gebruik gestandaardiseerde detectieregels** (MITRE/Wazuh)
- **Evalueer periodiek** de effectiviteit van detectieregels en logdekking
- **Houd rekening met compliance** en dataminimalisatie

- **Ondersteun kennisontwikkeling** bij betrokken medewerkers
- **Start kleinschalig** en breid uit via PoC en PDCA-cyclus

## Literatuurlijst (selectie)

- MITRE ATT&CK® (2025)
- Van der Tak, Kaya, Kara, Westerweel, & Onay (2025)
- ABN AMRO (2023)
- Ministerie van Justitie en Veiligheid (2024)
- NCSC (2024)
- CIS Controls v8 (2021)
- Wazuh documentatie (diverse links)

 [Edit this page](#)

# Installatie en Configuratie

## Stap 1: Repository klonen

Kloon de Wazuh-repository naar je systeem:

```
git clone https://github.com/wazuh/wazuh-docker.git -b v4.12.0
```

Ga vervolgens naar de `single-node` directory om alle hieronder beschreven commando's binnen deze directory uit te voeren.

## Stap 2: Certificaten regelen

Om communicatie tussen de nodes te beveiligen, zijn certificaten vereist. Je hebt twee mogelijkheden om deze certificaten te leveren:

### 2.1 Zelfondertekende certificaten genereren

Er is een standaard Docker image beschikbaar om het genereren van certificaten te automatiseren met behulp van de Wazuh certs gen tool.

Als je systeem een proxy gebruikt, voeg dan het volgende toe aan het `generate-indexer-certs.yml` bestand. Zo niet, sla deze stap dan over:

```
environment:  
  - HTTP_PROXY=YOUR_PROXY_ADDRESS_OR_DNS
```

Een volledig voorbeeld ziet er als volgt uit:

```
# Wazuh App Copyright (C) 2017 Wazuh Inc. (License GPLv2)  
version: '3'  
  
services:  
  generator:  
    image: wazuh/wazuh-certs-generator:0.0.2  
    hostname: wazuh-certs-generator
```

```
volumes:
  - ./config/wazuh_indexer_ssl_certs/:/certificates/
  - ./config/certs.yml:/config/certs.yml
environment:
  - HTTP_PROXY=YOUR_PROXY_ADDRESS_OR_DNS
```

Voer het volgende commando uit om de gewenste certificaten te verkrijgen:

```
docker-compose -f generate-indexer-certs.yml run --rm generator
```

Hiermee worden de certificaten opgeslagen in de map `config/wazuh_indexer_ssl_certs`.

## 2.2 Eigen certificaten plaatsen

Als je je eigen certificaten hebt, plaats ze dan als volgt in de map `config/wazuh_indexer_ssl_certs`:

### Wazuh indexer:

- `config/wazuh_indexer_ssl_certs/root-ca.pem`
- `config/wazuh_indexer_ssl_certs/wazuh.indexer-key.pem`
- `config/wazuh_indexer_ssl_certs/wazuh.indexer.pem`
- `config/wazuh_indexer_ssl_certs/admin.pem`
- `config/wazuh_indexer_ssl_certs/admin-key.pem`

### Wazuh manager:

- `config/wazuh_indexer_ssl_certs/root-ca-manager.pem`
- `config/wazuh_indexer_ssl_certs/wazuh.manager.pem`
- `config/wazuh_indexer_ssl_certs/wazuh.manager-key.pem`

### Wazuh dashboard:

- `config/wazuh_indexer_ssl_certs/wazuh.dashboard.pem`
- `config/wazuh_indexer_ssl_certs/wazuh.dashboard-key.pem`
- `config/wazuh_indexer_ssl_certs/root-ca.pem`

Als je de certificaten hebt gegenereerd of geplaatst, ga dan verder met **Stap 3** om de Wazuh single-node implementatie te starten.



# Stap 3: Start de Wazuh single-node implementatie

Start de Wazuh single-node implementatie met behulp van docker-compose:

## Voorgond:

```
docker-compose up
```

## Achtergrond:

```
docker-compose up -d
```

De standaard gebruikersnaam en het wachtwoord voor het Wazuh-dashboard zijn `admin` en `SecretPassword`. Voor extra beveiliging kun je het standaardwachtwoord van de Wazuh indexer admin-gebruiker wijzigen.

 [Edit this page](#)

# Onderzoeksresultaten

## Onderzoeksrapport: Gratis SIEM-oplossing voor het mkb

### Managementsamenvatting

- Het mkb is steeds vaker doelwit van cyberaanvallen, maar beschikt vaak niet over voldoende kennis of middelen.
- Commerciële SIEM-oplossingen zijn duur en complex.
- **Wazuh** is na vergelijking de aanbevolen gratis SIEM-oplossing voor het mkb: gebruiksvriendelijk, schaalbaar, goede communitysupport.
- Alternatieven zoals **Splunk Free** en **Elastic Security** zijn óf te beperkt, óf te complex zonder specialistische kennis.
- Implementatie van Wazuh wordt aanbevolen in fases, met externe ondersteuning waar nodig.

### Methodologie

- **Fases:**
  - i. Literatuurstudie
  - ii. Productanalyse (11 oplossingen → shortlist van 3)
  - iii. Productevaluatie met **SANS-matrix**
  - iv. Proof of Concept voorbereiding
- Criteria: functionaliteit, schaalbaarheid, onderhoud, juridische kaders.

### Vergelijking SIEM-oplossingen

Tool	Sterkte	Zwakte
Wazuh	Volledig open-source, schaalbaar, compliance	Initiële setup vereist tijd en kennis

Tool	Sterkte	Zwakte
	met NIS2 & AVG	
<b>Splunk Free</b>	Gebruiksvriendelijk, goede reputatie	Loglimiet van 500 MB/dag, geen multi-user
<b>Elastic SIEM</b>	Flexibel, uitbreidbaar (ELK-stack)	Hoge leercurve, complex beheer zonder IT-team

- **Wazuh scoorde het hoogst** op de aangepaste **SANS-matrix** ( $\pm 120$  evaluatiepunten).
- SWOT-analyse bevestigde geschiktheid voor mkb.

## Detectiescenario's en logbronnen

### Essentiële logbronnen:

1. Authenticatie- en toegangslogs (b.v. password spraying, beheerdersgroep-mutaties)
  2. Endpointbeveiligingslogs (b.v. ongewenste software, uitschakelen antivirus)
  3. Firewall- en netwerkklogs (b.v. C2-verkeer, datalekken)
- Elk type is gekoppeld aan concrete aanvalsscenario's die met Wazuh gedetecteerd kunnen worden.

## Schaalbaarheid en onderhoud

- **Technisch:** horizontaal & verticaal schalen; cloud-schaalbaarheid theoretisch mogelijk maar buiten scope.
- **Organisatorisch:** duidelijke rolverdeling, training, managementondersteuning.
- **Best practices:** gefaseerde uitrol (PDCA), logging beperken tot relevante data, automatisering, gebruiksvriendelijkheid.

## Werking van Wazuh

- **Verzameling:** via agents of agentless logging
- **Aggregatie:** normalisatie & opslag in Wazuh Indexer
- **Correlatie:** via regelsets, o.a. MITRE ATT&CK

- **Waarschuwingen:** alerts met prioriteitsniveaus (0–15), verschillende meldingsopties (e-mail, Slack, etc.).

## Niet-functionele eisen

- **Performance:** real-time detectie bij voldoende hardware
- **Flexibiliteit:** eenvoudig uit te breiden
- **Beveiliging:** encrypted communicatie, tamper detection, RBAC
- **Gebruiksgemak:** webdashboard, standaardregels, eenvoudige rapportage.

## Conclusie

Wazuh is de meest geschikte SIEM-oplossing voor het mkb dankzij functionaliteit, schaalbaarheid en gebruiksgemak. Ondanks een hogere initiële setup-inspanning biedt het langdurige voordelen.

## Advies

1. Start klein met beperkte logbronnen en detectiescenario's
2. Breid uit met meer bronnen & verfijnde regels
3. Monitor & optimaliseer beheerstructuur

Overweeg externe expertise indien IT-beveiliging in-house ontbreekt. PoC vormt de volgende stap.



[Edit this page](#)



# Technisch Ontwerp - Project Cyber Security

## Inleiding

### Doel

Dit document vormt het technische ontwerp voor de proof of concept "Gratis SIEM voor het mkb" op basis van Wazuh. Het doel is om:

- Componenten en infrastructuur technisch te definiëren
- Logbronnen, regels en validatie te beschrijven
- Toegangsbeheer te structureren
- Richtlijnen voor beheer en veiligheid te bieden

### Doelgroep

Systeem- en netwerkbeheerders binnen het mkb en externe consultants.

### Technische uitgangspunten

- Ubuntu 22.04 LTS (8 vCPU, 8GB RAM, 100GB opslag)
- Intern netwerkcommunicatie
- Agents via poort 1514 TCP
- Syslog via TCP 514
- TLS 1.2+
- Wazuh 4.12+

---

## Usecases

\*Voorbeeld van een uitgewerkte usecase. Het volledige Technisch ontwerp is op te vragen bij de projectgroep

# Detectie van credential attacks op Windows

- **Rule IDs:** 100010, 100011, 100012, 100013
- **Gedrag:**
  - LSASS dump via rundll32.exe / procdump
  - Credential Manager toegang via keymgr.dll
  - Vaultcmd uitlezing

## Scenario: Password spraying

In dit scenario worden pogingen tot het ongeautoriseerd verkrijgen en gebruiken van inloggegevens gedetecteerd, met nadruk op pogingen waarbij een beperkt aantal veelgebruikte wachtwoorden wordt getest op meerdere gebruikersaccounts (password spraying). Deze aanvalsmethode wordt vaak ingezet om lockouts te vermijden en langere tijd onopgemerkt te blijven.

### Componenten

- **Endpoints:** Windows 11
- **Wazuh-agent:** Geïnstalleerd op de endpoints
- **Wazuh Manager:** Ontvangt en analyseert loggegevens

### Detectie en regels

Gedrag	Bron	Rule ID	Beschrijving
Herhaalde foutieve logins op meerdere accounts	Medium	104390	Detecteert password spraying op basis van mislukte loginpogingen over accounts

### Verloop van scenario

1. Een aanvaller voert inlogpogingen uit met bekende wachtwoorden op meerdere gebruikers.
  2. Windows genereert bij elke mislukte poging een event met ID.
  3. De Wazuh-agent verzamelt deze logs.
  4. De Wazuh Manager vergelijkt de logs met ingestelde regels die patronen herkennen.
  5. Bij een match wordt een alert gegenereerd.
  6. Deze alert is zichtbaar in het Wazuh-dashboard voor verdere analyse.
- Herhaalde foutieve logins op meerdere accounts

\*Verdere usecases uit het technsich ontwerp. \*

## Detectie van malware

- **Rule ID:** 62123
- **Gedrag:**
  - Windows Defender detecteert mogelijk malware

## Toevoeging aan administratorsgroep

- **Rule ID:** 60160
- **Gedrag:**
  - Toevoegen van gebruiker aan admin-groep

## Bruteforce op Windows 11

- **Rule IDs:** 60122, 100100
- **Gedrag:**
  - Mislukte loginpogingen met verkeerde gebruikersnaam/wachtwoord

## Wissen van de Windows Logs

- **Rule ID:** 63104
- **Gedrag:**
  - Event ID 104 / 1102

## Uitschakelen van Windows Defender

- **Rule ID:** 62152
- **Gedrag:**
  - Windows Defender uitgeschakeld / niet actief

## Detectie van persistente aanvallen via Runkey-aanpassingen

- **Rule ID:** 598
- **Gedrag:**
  - Toevoegen van register runkey in HKLM/HKCU

## **Detectie van PowerShell-gebruik**

- **Rule ID:** 67027
- **Gedrag:**
  - Activatie van PowerShell-proces

## **Uitschakelen van Windows Firewall**

- **Rule ID:** 67005
- **Gedrag:**
  - Windows Firewall uitgeschakeld

## **Datacollectie van Active Directory met PowerShell**

- **Rule ID:** 100206
- **Gedrag:**
  - Powershell commando: Get-ADGroupMember

## **Inlogpoging op Windows 11 buiten werktijden**

- **Rule ID:** 60106
- **Gedrag:**
  - Login tussen 18:00 - 06:00

## **Inlogpoging op Windows 11 vanaf vreemd netwerk**

- **Rule ID:** 60106
- **Gedrag:**



- Login vanaf verdacht IP-adres

## Misbruik van de Windows Task Scheduler


- **Rule ID:** 92154
- **Gedrag:**
  - Proces laadt taskschd.dll

## Verdachte timestomping-activiteit

- **Rule IDs:** 61604, 101101
- **Gedrag:**
  - Aanmaaktijd van bestand gewijzigd

## Verdachte scriptingbestanden in /tmp of /gebruikers folder

- **Rule ID:** 92200
- **Gedrag:**
  - Aanmaak van .bat, .cmd, .ps1, etc. in tijdelijke/gebruikersmap

 [Edit this page](#)

# Testen van de Usecases

**NOTE:** De logging moet via audit policy toegestaan zijn voor verschillende Windows usecases, zoals beschreven in het Functioneel ontwerp

## Usecase Wissen van de Windows Logs

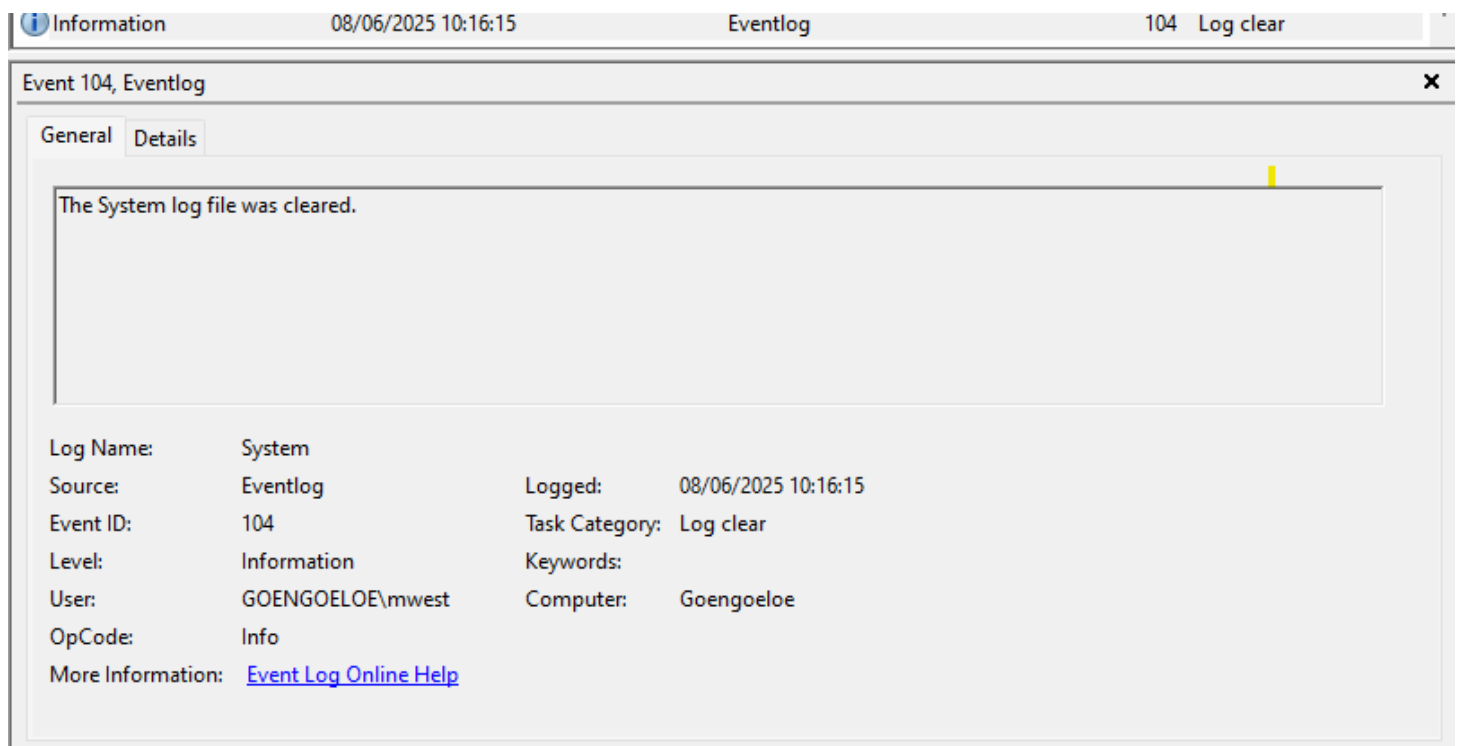
Uit te voeren met het volgende commmando in PowerShell met beheerrechten:

Voor het wissen van Systeem logs: `Clear-EventLog -LogName System`

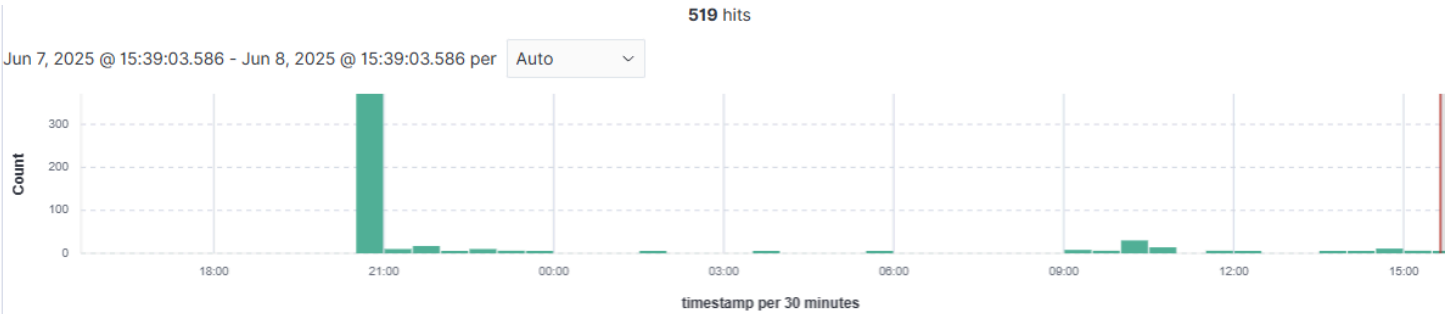
Voor het wissen van Security logs: `Clear-EventLog -LogName Security`

In de event viewer is het event gelijk te zien en een paar seconden later in Wazuh alert viewer:

### Event Viewer:



### Wazuh Alert:



05515 data.win.system.threadID: 9336 data.win.system.computer: Goengoeloe data.win.system.task: 104 data.win.system.processID: 3872

Expanded document [View surrounding documents](#) [View single document](#)

Table	JSON
<span>†</span> <span>_index</span>	wazuh-alerts-4.x-2025.06.08
<span>†</span> <span>agent.id</span>	004
<span>†</span> <span>agent.ip</span>	127.0.0.1
<span>†</span> <span>agent.name</span>	GONGOELOE
<span>†</span> <span>data.win.logFileCleared.channel</span>	System
<span>†</span> <span>data.win.logFileCleared.clientProcessId</span>	20376
<span>†</span> <span>data.win.logFileCleared.clientProcessStartKey</span>	28991922601202666
<span>†</span> <span>data.win.logFileCleared.subjectDomainName</span>	GOENGOELOE
<span>†</span> <span>data.win.logFileCleared.subjectUserName</span>	mwest
<span>†</span> <span>data.win.system.channel</span>	System
<span>†</span> <span>data.win.system.computer</span>	Goengoeloe
<span>†</span> <span>data.win.system.eventID</span>	104
<span>†</span> <span>data.win.system.eventRecordID</span>	105515
<span>†</span> <span>data.win.system.keywords</span>	0x8000000000000000
<span>†</span> <span>data.win.system.level</span>	4
<span>†</span> <span>data.win.system.message</span>	"The System log file was cleared."
<span>†</span> <span>data.win.system.opcode</span>	0
<span>†</span> <span>data.win.system.processID</span>	3872
<span>†</span> <span>data.win.system.providerGuid</span>	{fc65ddd8-d6ef-4962-83d5-6e5cfe9ce148}
<span>†</span> <span>data.win.system.providerName</span>	Microsoft-Windows-Eventlog

# Usecase Detectie van PowerShell-gebruik

Uit te voeren met het volgende commmando in PowerShell met beheerrechten:

Voor het activeren van Powershell: Start-Process Powershell.exe

In de event viewer is het event gelijk te zien en een paar seconden later in Wazuh alert viewer:

## Wazuh Alert:

PS C:\Users\mwest> Start-Process "powershell.exe"

PS C:\Users\mwest> |

ows\System32\WindowsPowerShell\v1.0\powershell.exe data.win.eventdata.targetLogonId: 0x0 data.win.eventdata.subjectUserSid: S-1

Expanded document

View surrounding documents

View single document

Table

JSON

t _index	wazuh-alerts-4.x-2025.06.09
t agent.id	004
t agent.ip	127.0.0.1
t agent.name	GONGOELOE
t data.win.eventdata.commandLine	"C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe"
t data.win.eventdata.mandatoryLabel	S-1-16-12288
t data.win.eventdata.newProcessId	0xba4
t data.win.eventdata.newProcessName	C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe
t data.win.eventdata.parentProcessName	C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe
t data.win.eventdata.processId	0x4f98
t data.win.eventdata.subjectDomainName	GOONGOELOE
t data.win.eventdata.subjectLogonId	0x61282
t data.win.eventdata.subjectUserName	mwest
t data.win.eventdata.subjectUserSid	S-1-5-21-370239534-428301900-2303973759-1001
t data.win.eventdata.targetLogonId	0x0
t data.win.eventdata.targetUserSid	S-1-0-0

# Usecase Detectie van persistente aanvallen via Windows Registry Runkeys

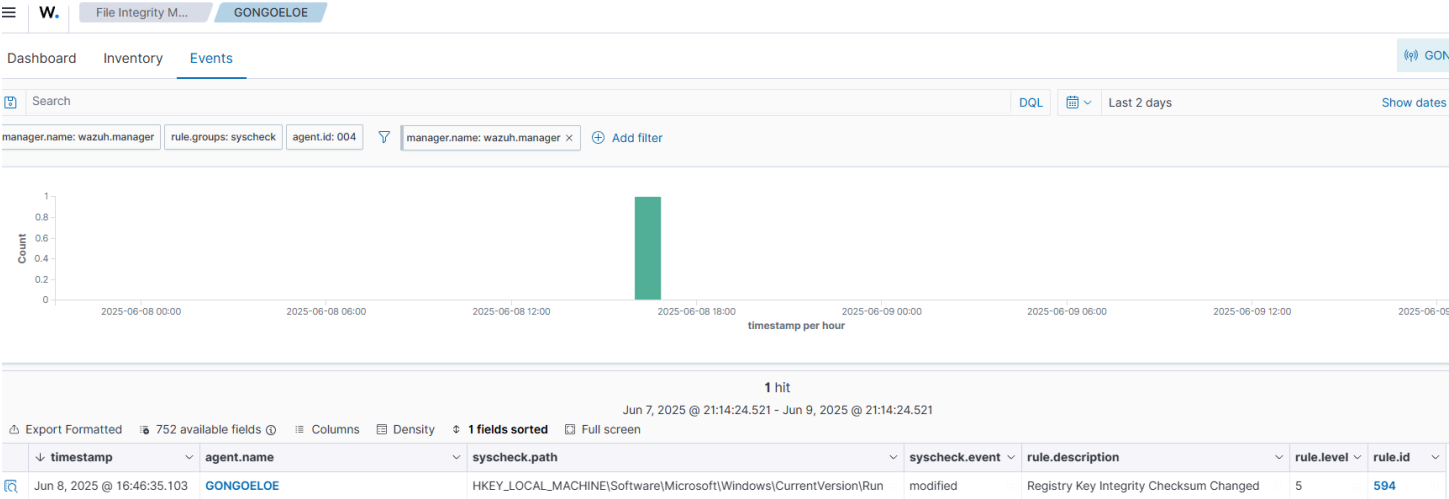
Uit te voeren met het volgende commmando in PowerShell met beheerrechten:

Voor het activeren van Powershell:

```
New-ItemProperty -Path "HKCU:\Software\Microsoft\Windows\CurrentVersion\Run" -Name "TestPersister" -Value "C:\Windows\System32\notepad.exe" -PropertyType "String"
```

In de event viewer is het event gelijk te zien en een FIM system check later in Wazuh alert viewer en FIM module viewer:

**Wazuh Alert:**



Wazuh FIM events:

# Usecase Toevoegen aan de administratorsgroep

Maak eerst een testgebruiker aan met de volgende commando:

```
New-LocalUser -Name "testuser" -Password (ConvertTo-SecureString "Heelgoedenveilig!!!" -AsPlainText -Force)
```

Voeg een gebruiker aan de administrator groep:

```
Add-LocalGroupMember -Group "Administrators" -Member "testuser"
```

Nu zie je dat er een high alert is op de wazuh dashboard:

**Wazuh Alert:**

## Usecase Detectie van credential attacks op Windows

Om nu de usecase te testen kan je een van de volgende commando's op de powershell van de Windows systeem uitvoeren:

```
rundll32.exe C:\Windows\System32\comsvcs.dll, MiniDump 624 C:\temp\lsass.dmp full  
rundll32 keymgr.dll,KRShowKeyMgr  
vaultcmd /listcreds:"Windows Credentials" /all
```

**Wazuh Alert:**







# Usecase Detectie van Malware

Om nu de usecase te testen voer het volgende commando op de powershell van de Windows systeem uitvoeren:

```
Set-Content "$env:USERPROFILE\Desktop\eicar.com"  
'X5O!P%@AP[4\PZX54(P^)7CC)7}$EICAR-STANDARD-ANTIVIRUS-TEST-FILE!$H+H*'
```

**Wazuh Alert:**

# Usecase Uitschakelen van Windows Defender

Voer het volgende commando uit in Powershell om een `.bat`-bestand aan te maken in de `TEMP`-folder:

```
echo "echo hello" > "$env:TEMP\Hacked_me.bat"
```

Vervolgens genereert Sysmon een melding in de eventviewer met ID 11:

**Windows Event\_viewer:**

**Wazuh Event\_viewer:**

## Usecase Misbruik van de Windows Task Scheduler

Voer het volgende commando uit in Powershell om het taskschd.dll module te laden:

```
Start-Process schtasks.exe
```

Vervolgens genereert Sysmon een melding in de eventviewer met ID 7:

**Windows Event\_viewer:**

**Wazuh Event\_viewer:**

## **Usecase Bruteforce op Windows 11**

Vergrendel het Windows apparaat waar de Wazuh agent zich op bevind en voer herhaaldelijk een verkeerd wachtwoord in. Na 3x binnen 2 minuten wordt er een melding gegenereerd

**Windows Event\_viewer:**

**Wazuh Event\_viewer:**

## **Usecase Uitschakelen van Windows Firewall**

Voer het volgende PowerShell-commando uit:

```
Set-NetFirewallProfile -Profile Domain,Public,Private -Enabled False
```

**Wazuh Event\_viewer:**

## Usecase AD activiteiten scannen (SharpHound)

Voer het volgende PowerShell-commando uit (met SharpHound geïnstalleerd) in de folder van SharpHound:

```
.\SharpHound.exe --CollectionMethods All -Loop
```

**Wazuh Event\_viewer:**

# Usecase Inlogpoging op Windows 11 buiten werktijden

Log in het Windows endpoint buiten kantoortijden (18:30-08:30)

**Wazuh Event\_viewer:**

# Usecase Inlogpoging op Windows 11 vanaf onbekend netwerk

Log in op een Windows client vanaf een 192.168.56.0/24 netwerk (gebruik rdp)

**Wazuh Event\_viewer:**

## Usecase Password Spraying

Probeer 5x om in te loggen met verkeerde wachtwoorden van zelfde ip address

**Wazuh Event\_viewer:**



## Usecase Verdachte scripting bestanden in `/tmp` of `/gebruikers` folder

Voer het volgende commando uit in Powershell om een .bat bestand aan te maken in de TEMP folder:

```
echo "echo hello" > "$env:TEMP\Hacked_me.bat"
```

**Wazuh Event\_viewer:**

 [Edit this page](#)

# Installatie en Deploy van de Wazuh Windows-Agent

Deze handleiding beschrijft stap voor stap hoe je de Wazuh-agent installeert en configureert op een Windows-machine, zodat deze verbinding maakt met een lokaal draaiende Wazuh-server (bijvoorbeeld een single-node Docker-implementatie).

## 1. Vereisten

- **Windows-versie:** Windows 10 (Pro/Enterprise) of Windows Server 2016/2019/2022
- **Administrator-rechten:** Open PowerShell als Administrator om de MSI te installeren
- **PowerShell 3.0 of hoger:** Controleer met:

```
$PSVersionTable.PSVersion
```

\* **\*\*Netwerk\*\*:** De Wazuh-manager draait op dezelfde host, bereikbaar via ``localhost`` (127.0.0.1)

\* **\*\*Poorten opengezet\*\*:**

\* UDP 1514 (log-verkeer)

\* TCP 1515 (agent-registratie)

\* TCP 443 (Kibana/Wazuh-dashboard)

Zorg dat Windows Defender Firewall of een andere firewall deze poorten toestaat (zie sectie 4)

### ## 2. Installatiepakket kiezen

Wazuh biedt voor Windows een MSI-installer aan. Kies in de UI of op de downloadpagina het juiste installatiepakket:

Architectuur	Installatiebestand
-----	-----
32-bit / 64-bit Intel (x86/x64)	MSI - Intel
Apple Silicon (M1/M2, etc.)	MSI - Apple Silicon

Voorbeelden van beschikbare downloads (versie 4.12.0):

```
* **Intel MSI**:  
`https://packages.wazuh.com/4.x/windows/wazuh-agent-4.12.0-1.msi`  
  
* **Apple Silicon MSI**:  
`https://packages.wazuh.com/4.x/windows/wazuh-agent-4.12.0-1-arm64.msi`
```

In dit document gaan we ervan uit dat je de **Intel (32/64-bit) MSI** (versie 4.12.0) downloadt.

## ## 3. Download en installatie

### 1. Open PowerShell als Administrator

\* Klik op Start, typ "PowerShell", rechtsklik op "Windows PowerShell" en kies **Als administrator uitvoeren**.

### 2. Download de Wazuh-agent MSI

Navigeer naar een tijdelijke map (bijvoorbeeld je Downloads-folder) en voer het volgende commando uit:

```
`powershell  
cd ~\Downloads  
Invoke-WebRequest -Uri "https://packages.wazuh.com/4.x/windows/wazuh-agent-4.12.0-1.msi" `  
-OutFile "wazuh-agent-4.12.0-1.msi"
```

### 3. Installeer de Wazuh-agent Vervang in dit commando:

- `127.0.0.1` → het IP of de FQDN van je Wazuh-manager
- `MyAgentName` → een unieke naam voor de agent (mag geen spaties bevatten)
- `Default` → de groep waarin deze agent wordt geplaatst

```
msiexec.exe /i "wazuh-agent-4.12.0-1.msi" `  
/qn `  
WAZUH_MANAGER="127.0.0.1" `  
WAZUH_AGENT_NAME="MyAgentName" `  
WAZUH_AGENT_GROUPS="Default" `  
/l*v "C:\Windows\Temp\wazuh-agent-install.log"
```

- `/qn` → stille installatie zonder GUI
- `/l*v ...` → schrijft een uitgebreid installatie-log naar Windows Temp

## 4. (Optioneel) MSI via GUI

- Dubbelklik op `wazuh-agent-4.12.0-1.msi`.
- Bij **Server address** voer je in: `127.0.0.1`
- Onder **Assign an agent name** geef je een unieke naam (bijvoorbeeld `MyAgentName`).
- Onder **Select one or more existing groups** kies je "Default" (of een andere groep).
- Klik op **Install** en wacht tot de installatie is voltooid.

## 4. Firewallconfiguratie

Zorg dat de agent met de manager kan communiceren door deze regels in Windows Defender Firewall in te stellen:

1. Open **Windows Defender Firewall met geavanceerde beveiliging** (via Start → typ "Windows Defender Firewall...").
2. Ga naar **Inkomende regels** → **Nieuwe regel...** → kies **Poort** → klik **Volgende**.
3. Selecteer **UDP**, vul **1514** in, klik **Volgende** → kies **Verbinding toestaan** → klik **Volgende**.
4. Selecteer de gewenste profielen (Domain, Private, Public) → klik **Volgende**.
5. Geef de regel de naam:

Wazuh UDP 1514

Klik op **Voltooien**.

6. Herhaal stap 2 t/m 5 voor **TCP-poorten 1515 en 443**:

- Kies **Poort** → **TCP** → "1515,443" → **Verbinding toestaan** → kies profielen → naam

Wazuh TCP 1515 & 443

7. Ga naar **Uitgaande regels** → maak dezelfde twee regels aan (UDP 1514 en TCP 1515/443) zodat eventuele outbound-restricties ook worden opgeheven.

## 5. De agent starten en controleren

1. **Start de Wazuh-agent (indien nodig)**

```
net start wazuh-agent
```

Je ziet:

```
The Wazuh Agent service was started successfully.
```

## 2. Controleer in Services

- Open **Services** (`services.msc`).
- Zoek naar **Wazuh Agent**. De status moet "Running" zijn.

## 3. Bekijk de agentlog voor fouten

```
cd "C:\Program Files\Wazuh\logs"  
type ossec.log
```

Je zou regels moeten zien zoals:

```
2025/06/01 13:45:12 ossec-agentd: INFO: (6001): Started agent-based daemon.  
2025/06/01 13:45:15 ossec-agentd: INFO: (2042): Connected to the Wazuh manager  
(127.0.0.1:1515).
```

- Als er fouten staan (zoals "Connection refused" of "Timeout"), controleer firewall en of de manager in Docker draait.

## 4. Verifieer in het Wazuh-dashboard (Kibana)

- Open je browser en ga naar:

```
https://localhost
```

- Log in op het Wazuh-dashboard → navigeer naar **Wazuh** → **Agents**.
- De Windows-agent moet hier met status **Active** verschijnen.

# 6. Veelvoorkomende issues en troubleshooting

- **Agent verschijnt niet in Dashboard**

i. Herstart de agentdienst:

```
Restart-Service wazuh-agent
```

ii. Controleer in Docker dat de manager actief is:

```
docker ps
```

- Zoek naar `single-node-wazuh.manager-1`. De status moet **Up** zijn en de poorten 1514/1515 open hebben.

iii. Bekijk de manager-logs op connecties:

```
docker exec -it single-node-wazuh.manager-1 /bin/bash  
tail -n 20 /var/ossec/logs/ossec.log
```

- Zoek naar regels zoals "New agent connection from 127.0.0.1".

- **UDP-1514 niet bereikbaar**

- Controleer de firewallregels (zie sectie 4).
- Test of de Wazuh-manager-container op UDP 1514 luistert:

```
netstat -an | findstr 1514
```

- Je zou iets moeten zien als:

```
UDP      0.0.0.0:1514      *:*
```

- Voer de UDP-test in PowerShell uit:

```
$udpClient = New-Object System.Net.Sockets.UdpClient  
$endpoint = New-Object  
System.Net.IPEndPoint([System.Net.IPAddress]::Parse("127.0.0.1"), 1514)  
$message = [System.Text.Encoding]::ASCII.GetBytes("UDP Test")  
$bytesSent = $udpClient.Send($message, $message.Length, $endpoint)
```

```
$udpClient.Close()  
Write-Host "Verstuurd $bytesSent bytes naar UDP 1514"
```

- Controleer in de manager-logs of het pakket aankomt.

- **Verkeerd serveradres ingevoerd**

- Tijdens MSI-installatie kan het adres niet achteraf worden gewijzigd.
- Verwijder de agent, verwijder de map `C:\Program Files\Wazuh`, en installeer opnieuw met het juiste adres.

 [Edit this page](#)



# Dashboard Toegang en Inloggen

Na installatie en configuratie van de Wazuh-manager en -agenten kun je inloggen op het Wazuh-dashboard (Kibana/Wazuh App). Hieronder staat hoe je dat doet met de standaard (default) inloggegevens en hoe je, indien gewenst, deze later kunt wijzigen.

## Toegang krijgen tot het Dashboard

### 1. URL openen

- Als je de standaard single-node Docker-implementatie gebruikt, draait het dashboard op poort 443 van de Windows-host.
- Open je browser en ga naar:

```
https://localhost (127.0.0.1)
```

(of vervang "localhost" door het IP-adres of de FQDN van de host als je dat anders hebt ingesteld).

### 2. Security-waarschuwing accepteren (indien van toepassing)

- Omdat Docker-containers doorgaans met een zelfondertekend certificaat werken, zal je browser mogelijk een "Niet beveiligde verbinding" of "Onbetrouwbaar certificaat" melding geven.
- Klik in dat geval op "Geavanceerd" (of "Advanced") → "Verder naar localhost (onveilig)" (of "Proceed to localhost (unsafe)") om door te gaan.


## Default Inloggegevens

Bij een verse installatie van de Wazuh single-node Docker-stack (versie 4.12.0) gelden de volgende standaardgebruikersnamen en wachtwoorden:

- **Wazuh-dashboard (Kibana + Wazuh App)**

- **Gebruikersnaam:** `admin`

◦ **Wachtwoord:**

 [Edit this page](#)

# Deploy Windows Agent

Deze handleiding beschrijft stap voor stap hoe je de Wazuh-agent installeert en configureert op een Windows-machine, zodat deze verbinding maakt met een lokaal draaiende Wazuh-server (bijvoorbeeld een single-node Docker-implementatie).

(Volledige inhoud van het originele bestand volgt hieronder)

## Installatie en Deploy van de Wazuh Windows-Agent

Deze handleiding beschrijft stap voor stap hoe je de Wazuh-agent installeert en configureert op een Windows-machine, zodat deze verbinding maakt met een lokaal draaiende Wazuh-server (bijvoorbeeld een single-node Docker-implementatie).

### 1. Vereisten

- **Windows-versie:** Windows 10 (Pro/Enterprise) of Windows Server 2016/2019/2022
- **Administrator-rechten:** Open PowerShell als Administrator om de MSI te installeren
- **PowerShell 3.0 of hoger:** Controleer met:

```
$PSVersionTable.PSVersion
```

- **Netwerk:** De Wazuh-manager draait op dezelfde host, bereikbaar via `localhost` (127.0.0.1)
- **Poorten opengezet:**
  - UDP 1514 (log-verkeer)
  - TCP 1515 (agent-registratie)
  - TCP 443 (Kibana/Wazuh-dashboard) Zorg dat Windows Defender Firewall of een andere firewall deze poorten toestaat (zie sectie 4)

### 2. Installatiepakket kiezen

Wazuh biedt voor Windows een MSI-installer aan. Kies in de UI of op de downloadpagina het juiste installatiepakket:

Architectuur	Installatiebestand
32-bit / 64-bit Intel (x86/x64)	MSI – Intel
Apple Silicon (M1/M2, etc.)	MSI – Apple Silicon

Voorbeelden van beschikbare downloads (versie 4.12.0):

- **Intel MSI:** `https://packages.wazuh.com/4.x/windows/wazuh-agent-4.12.0-1.msi`
- **Apple Silicon MSI:** `https://packages.wazuh.com/4.x/windows/wazuh-agent-4.12.0-1-arm64.msi`

In dit document gaan we ervan uit dat je de **Intel (32/64-bit) MSI** (versie 4.12.0) downloadt.

## 3. Download en installatie

### 1. Open PowerShell als Administrator

- Klik op Start, typ "PowerShell", rechtsklik op "Windows PowerShell" en kies **Als administrator uitvoeren**.

### 2. Download de Wazuh-agent MSI

Navigeer naar een tijdelijke map (bijvoorbeeld je Downloads-folder) en voer het volgende commando uit:

```
cd ~\Downloads
Invoke-WebRequest -Uri "https://packages.wazuh.com/4.x/windows/wazuh-agent-4.12.0-1.msi" `
    -OutFile "wazuh-agent-4.12.0-1.msi"
```

### 3. Installeer de Wazuh-agent

Vervang in dit commando:

- `127.0.0.1` → het IP of de FQDN van je Wazuh-manager
- `MyAgentName` → een unieke naam voor de agent (mag geen spaties bevatten)
- `Default` → de groep waarin deze agent wordt geplaatst

```
msiexec.exe /i "wazuh-agent-4.12.0-1.msi" `
    /qn `
```

```
WAZUH_MANAGER="127.0.0.1" `
WAZUH_AGENT_NAME="MyAgentName" `
WAZUH_AGENT_GROUPS="Default" `
/l*v "C:\Windows\Temp\wazuh-agent-install.log"
```

- `/qn` → stille installatie zonder GUI
- `/l*v ...` → schrijft een uitgebreid installatie-log naar Windows Temp

#### 4. (Optioneel) MSI via GUI

- Dubbelklik op `wazuh-agent-4.12.0-1.msi`.
- Bij **Server address** voer je in: `127.0.0.1`
- Onder **Assign an agent name** geef je een unieke naam (bijvoorbeeld `MyAgentName`).
- Onder **Select one or more existing groups** kies je "Default" (of een andere groep).
- Klik op **Install** en wacht tot de installatie is voltooid.

## 4. Firewallconfiguratie

Zorg dat de agent met de manager kan communiceren door deze regels in Windows Defender Firewall in te stellen:

1. Open **Windows Defender Firewall met geavanceerde beveiliging** (via Start → typ "Windows Defender Firewall...").
2. Ga naar **Inkomende regels** → **Nieuwe regel...** → kies **Poort** → klik **Volgende**.
3. Selecteer **UDP**, vul **1514** in, klik **Volgende** → kies **Verbinding toestaan** → klik **Volgende**.
4. Selecteer de gewenste profielen (Domain, Private, Public) → klik **Voltooien**.
5. Geef de regel de naam:

Wazuh UDP 1514

Klik op **Voltooien**.

6. Herhaal stap 2 t/m 5 voor **TCP-poorten 1515 en 443**:

- Kies **Poort** → **TCP** → "1515,443" → **Verbinding toestaan** → kies profielen → naam

7. Ga naar **Uitgaande regels** → maak dezelfde twee regels aan (UDP 1514 en TCP 1515/443) zodat eventuele outbound-restricties ook worden opgeheven.

## 5. De agent starten en controleren

### 1. Start de Wazuh-agent (indien nodig)

```
net start wazuh-agent
```

Je ziet:

```
The Wazuh Agent service was started successfully.
```

### 2. Controleer in Services

- Open **Services** (`services.msc`).
- Zoek naar **Wazuh Agent**. De status moet "Running" zijn.

### 3. Bekijk de agentlog voor fouten

```
cd "C:\Program Files\Wazuh\logs"  
type ossec.log
```

Je zou regels moeten zien zoals:

```
2025/06/01 13:45:12 ossec-agentd: INFO: (6001): Started agent-based daemon.  
2025/06/01 13:45:15 ossec-agentd: INFO: (2042): Connected to the Wazuh manager  
(127.0.0.1:1515).
```

- Als er fouten staan (zoals "Connection refused" of "Timeout"), controleer firewall en of de manager in Docker draait.

### 4. Verifieer in het Wazuh-dashboard (Kibana)

- Open je browser en ga naar:

```
https://localhost
```

- Log in op het Wazuh-dashboard → navigeer naar **Wazuh** → **Agents**.
- De Windows-agent moet hier met status **Active** verschijnen.

## 6. Veelvoorkomende issues en troubleshooting

- **Agent verschijnt niet in Dashboard**

i. Herstart de agentdienst:

```
Restart-Service wazuh-agent
```

ii. Controleer in Docker dat de manager actief is:

```
docker ps
```

- Zoek naar `single-node-wazuh.manager-1`. De status moet **Up** zijn en de poorten 1514/1515 open hebben.

iii. Bekijk de manager-logs op connecties:

```
docker exec -it single-node-wazuh.manager-1 /bin/bash  
tail -n 20 /var/ossec/logs/ossec.log
```

- Zoek naar regels zoals "New agent connection from 127.0.0.1".

- **UDP-1514 niet bereikbaar**

- Controleer de firewallregels (zie sectie 4).
- Test of de Wazuh-manager-container op UDP 1514 luistert:

```
netstat -an | findstr 1514
```

- Je zou iets moeten zien als:

UDP 0.0.0.0:1514

\*.\*

- Voer de UDP-test in PowerShell uit:

```
$udpClient = New-Object System.Net.Sockets.UdpClient
$endpoint = New-Object
System.Net.IPEndPoint([System.Net.IPAddress]::Parse("127.0.0.1"), 1514)
$message = [System.Text.Encoding]::ASCII.GetBytes("UDP Test")
$bytesSent = $udpClient.Send($message, $message.Length, $endpoint)
$udpClient.Close()
Write-Host "Verstuurd $bytesSent bytes naar UDP 1514"
```

- Controleer in de manager-logs of het pakket aankomt.

- **Verkeerd serveradres ingevoerd**

- Tijdens MSI-installatie kan het adres niet achteraf worden gewijzigd.
- Verwijder de agent, verwijder de map `C:\Program Files\Wazuh`, en installeer opnieuw met het juiste adres.

## 7. Veelgestelde vragen (FAQ)

1. **"Kan ik in plaats van `localhost` ook mijn LAN-IP invullen?"** Ja, als de Wazuh-manager in Docker via dat LAN-IP bereikbaar is (bijv. `192.168.1.10:1515`), kun je dit IP opgeven tijdens de installatie. Zorg dat de Docker-poortbinding juist is en dat de firewall dit LAN-verkeer toestaat.
2. **"Hoe geef ik meerdere groepen op?"** Tijdens stille installatie (MSI) kun je meerdere groepen comma-gescheiden opgeven:

```
msiexec.exe /i "wazuh-agent-4.12.0-1.msi" /qn `
WAZUH_MANAGER="127.0.0.1" `
WAZUH_AGENT_GROUPS="GroupA,GroupB"
```


3. **"Waar vind ik de agentconfiguratie na installatie?"** De configuratiebestanden staan in:

```
C:\Program Files\Wazuh\etc\ossec.conf
```

Pas hierin instellingen aan (bijvoorbeeld logging-levels), en herstart dan de dienst:



Restart-Service wazuh-agent

 [Edit this page](#)

# Docker Compose Configuratie

Hieronder vind je het gebruikte docker-compose.yaml bestand voor de Wazuh stack:

```
# Wazuh App Copyright (C) 2017, Wazuh Inc. (License GPLv2)
```

```
version: '3.7'
```

```
services:
```

```
  wazuh.manager:
```

```
    image: wazuh/wazuh-manager:4.12.0
```

```
    hostname: wazuh.manager
```

```
    restart: always
```

```
    ulimits:
```

```
      memlock:
```

```
        soft: -1
```

```
        hard: -1
```

```
      nofile:
```

```
        soft: 655360
```

```
        hard: 655360
```

```
    ports:
```

```
      - "1514:1514"
```

```
      - "1515:1515"
```

```
      - "514:514/udp"
```

```
      - "55000:55000"
```

```
    environment:
```

```
      - INDEXER_URL=https://wazuh.indexer:9200
```

```
      - INDEXER_USERNAME=admin
```

```
      - INDEXER_PASSWORD=SecretPassword
```

```
      - FILEBEAT_SSL_VERIFICATION_MODE=full
```

```
      - SSL_CERTIFICATE_AUTHORITIES=/etc/ssl/root-ca.pem
```

```
      - SSL_CERTIFICATE=/etc/ssl/filebeat.pem
```

```
      - SSL_KEY=/etc/ssl/filebeat.key
```

```
      - API_USERNAME=wazuh-wui
```

```
      - API_PASSWORD=MyS3cr37P450r.*-
```

```
    volumes:
```

```
      - wazuh_api_configuration:/var/ossec/api/configuration
```

```
      - wazuh_etc:/var/ossec/etc
```

```
      - wazuh_logs:/var/ossec/logs
```

```
      - wazuh_queue:/var/ossec/queue
```

```
      - wazuh_var_multigroups:/var/ossec/var/multigroups
```

```
      - wazuh_integrations:/var/ossec/integrations
```

```
      - wazuh_active_response:/var/ossec/active-response/bin
```

```
      - wazuh_agentless:/var/ossec/agentless
```

- wazuh\_wodles:/var/ossec/wodles
- filebeat\_etc:/etc/filebeat
- filebeat\_var:/var/lib/filebeat
- ./config/wazuh\_indexer\_ssl\_certs/root-ca-manager.pem:/etc/ssl/root-ca.pem
- ./config/wazuh\_indexer\_ssl\_certs/wazuh.manager.pem:/etc/ssl/filebeat.pem
- ./config/wazuh\_indexer\_ssl\_certs/wazuh.manager-key.pem:/etc/ssl/filebeat.key
- ./config/wazuh\_cluster/wazuh\_manager.conf:/wazuh-config-mount/etc/ossec.conf
- ./custom\_rules/local\_rules.xml:/var/ossec/etc/rules/local\_rules.xml:rw

#### wazuh.indexer:

```

image: wazuh/wazuh-indexer:4.12.0
hostname: wazuh.indexer
restart: always
ports:
  - "9200:9200"
environment:
  - "OPENSEARCH_JAVA_OPTS=-Xms1g -Xmx1g"
ulimits:
  memlock:
    soft: -1
    hard: -1
  nofile:
    soft: 65536
    hard: 65536
volumes:
  - wazuh-indexer-data:/var/lib/wazuh-indexer
  - ./config/wazuh_indexer_ssl_certs/root-ca.pem:/usr/share/wazuh-indexer/certs/root-ca.pem
  - ./config/wazuh_indexer_ssl_certs/wazuh.indexer-key.pem:/usr/share/wazuh-indexer/certs/wazuh.indexer.key
  - ./config/wazuh_indexer_ssl_certs/wazuh.indexer.pem:/usr/share/wazuh-indexer/certs/wazuh.indexer.pem
  - ./config/wazuh_indexer_ssl_certs/admin.pem:/usr/share/wazuh-indexer/certs/admin.pem
  - ./config/wazuh_indexer_ssl_certs/admin-key.pem:/usr/share/wazuh-indexer/certs/admin-key.pem
  - ./config/wazuh_indexer/wazuh.indexer.yml:/usr/share/wazuh-indexer/opensearch.yml
  - ./config/wazuh_indexer/internal_users.yml:/usr/share/wazuh-indexer/opensearch-security/internal_users.yml

```

#### wazuh.dashboard:

```

image: wazuh/wazuh-dashboard:4.12.0
hostname: wazuh.dashboard
restart: always
ports:
  - 443:5601

```

#### environment:

- INDEXER\_USERNAME=admin
- INDEXER\_PASSWORD=SecretPassword
- WAZUH\_API\_URL=https://wazuh.manager
- DASHBOARD\_USERNAME=kibanaserver
- DASHBOARD\_PASSWORD=kibanaserver
- API\_USERNAME=wazuh-wui
- API\_PASSWORD=MyS3cr37P450r.\*-

#### volumes:

- ./config/wazuh\_indexer\_ssl\_certs/wazuh.dashboard.pem:/usr/share/wazuh-dashboard/certs/wazuh-dashboard.pem
- ./config/wazuh\_indexer\_ssl\_certs/wazuh.dashboard-key.pem:/usr/share/wazuh-dashboard/certs/wazuh-dashboard-key.pem
- ./config/wazuh\_indexer\_ssl\_certs/root-ca.pem:/usr/share/wazuh-dashboard/certs/root-ca.pem
- ./config/wazuh\_dashboard/opensearch\_dashboards.yml:/usr/share/wazuh-dashboard/config/opensearch\_dashboards.yml
- ./config/wazuh\_dashboard/wazuh.yml:/usr/share/wazuh-dashboard/data/wazuh/config/wazuh.yml
- wazuh-dashboard-config:/usr/share/wazuh-dashboard/data/wazuh/config
- wazuh-dashboard-custom:/usr/share/wazuh-dashboard/plugins/wazuh/public/assets/custom

#### depends\_on:

- wazuh.indexer

#### links:

- wazuh.indexer:wazuh.indexer
- wazuh.manager:wazuh.manager

#### volumes:

wazuh\_api\_configuration:

wazuh\_etc:

wazuh\_logs:

wazuh\_queue:

wazuh\_var\_multigroups:

wazuh\_integrations:

wazuh\_active\_response:

wazuh\_agentless:

wazuh\_wodles:

filebeat\_etc:

filebeat\_var:

wazuh-indexer-data:

wazuh-dashboard-config:

wazuh-dashboard-custom:

# Local Rules (Wazuh)

Hieronder vind je de lokale regels voor Wazuh, zoals gebruikt in deze implementatie:

```
<group name="local">

  <!-- SSHD brute-force vanaf bekend IP (voorbeeld UNIX) -->
  <rule id="100001" level="5">
    <if_sid>5716</if_sid>
    <srcip>1.1.1.1</srcip>
    <description>sshd: authentication failed from IP 1.1.1.1.</description>
    <group>authentication_failed,pci_dss_10.2.4,pci_dss_10.2.5</group>
  </rule>

</group>

<group name="rdp">

  <!-- Herhaalde RDP-aanmeldingen -->
  <rule id="100002" level="10" frequency="3" timeframe="120">
    <if_matched_sid>60122</if_matched_sid>
    <description>RDP Attack Detected</description>
  </rule>

</group>

<group name="apache">

  <!-- Apache: verboden bestandspad geprobeerd -->
  <rule id="100003" level="5">
    <if_sid>30101</if_sid>
    <match>denied by server configuration</match>
    <description>Apache: Attempt to access forbidden file or directory.
  </description>
    <group>access_denied</group>
  </rule>

</group>

<group name="windows,windows_security">

  <!-- Password reset attempt (Event ID 4724) -->
  <rule id="100004" level="8">
```

```

    <if_sid>60103</if_sid>
    <field name="win.system.eventID">^4724$</field>
    <description>Attempt to reset password for: $(win.eventdata.TargetUserName).
</description>
    <options>no_full_log</options>
</rule>

<!-- Aanmaken van nieuw gebruikersaccount (Event ID 4720) -->
<rule id="100005" level="8">
    <field name="win.system.eventID">4720</field>
    <description>New user account created: $(win.eventdata.TargetUserName).
</description>
</rule>

<!-- Auditlog gewist (Event ID 1102) -->
<rule id="100006" level="10">
    <field name="win.system.eventID">1102</field>
    <description>The audit log was cleared. Mogelijke poging om sporen te wissen.
</description>
</rule>

</group>

<group name="windows,powershell">

    <!-- Start van PowerShell (Event ID 4688) -->
    <rule id="100007" level="10">
        <field name="win.system.eventID">4688</field>
        <field name="win.eventdata.NewProcessName">(?
i).*\powershell.exe$|.*\pwsh.exe$</field>
        <description>PowerShell-process gestart via Event ID 4688</description>
    </rule>

</group>

<group name="windows,registry">

    <!-- Wijziging aan Run registry key -->
    <rule id="100008" level="10">
        <if_sid>598</if_sid>
        <description>Nieuwe waarde toegevoegd aan Windows Run registry key. Mogelijke
persistente aanval.</description>
        <mitre>
            <id>T1547.001</id>
        </mitre>
    </rule>

</group>

```

```

<group name="Windows,attack,">
  <!-- Detecting an LSASS memory dumping attack using Rundll32.exe Minidump
Function or Comsvcs.dll Exploitation -->
  <rule id="100010" level="10">
    <if_sid>61609</if_sid>
    <field name="win.eventdata.image" type="pcre2">(?!i)\\\\rundll32.exe</field>
    <field name="win.eventdata.imageLoaded" type="pcre2">(?!i)[c-
z]:\\\\Windows\\\\System32\\\\comsvcs\\.dll</field>
    <description>Possible adversary activity - LSASS memory dump:
$(win.eventdata.imageLoaded) loaded by using $(win.eventData.image) on
$(win.system.computer).</description>
    <mitre>
      <id>T1003.001</id>
    </mitre>
  </rule>

  <!-- Detecting an LSASS memory dumping attack using specialized tools -->
  <rule id="100011" level="10">
    <if_sid>61613</if_sid>
    <field name="win.eventData.targetFilename" type="pcre2">(?!i)\\\\\\
[^\]\\\\*\\\\.dmp$</field>
    <field name="win.eventData.image" negate="yes" type="pcre2">(?!i)\\\\\\\\lsass.*
</field>
    <description>Possible adversary activity - LSASS memory dump:
$(win.eventdata.image) created a new file on $(win.system.computer) endpoint.
</description>
    <mitre>
      <id>T1003.001</id>
    </mitre>
  </rule>

  <!-- Detecting a Windows Credential Manager exploitation attack -->
  <rule id="100012" level="10">
    <if_sid>61603</if_sid>
    <field name="win.eventData.Image" type="pcre2">(?!i)\\\\rundll32.exe</field>
    <field name="win.eventData.commandLine"
type="pcre2">keymgr.dll,KRShowKeyMgr</field>
    <description>Possible adversary activity - Credential Manager Access via
$(win.eventData.Image) on $(win.system.computer) endpoint.</description>
    <mitre>
      <id>T1003</id>
    </mitre>
  </rule>

  <!-- Detecting a Windows Credential Manager exploitation attack by VaultCmd
process enumeration -->
  <rule id="100013" level="10">
    <if_sid>92052</if_sid>
    <field name="win.eventData.image" type="pcre2">(?!i)\\\\\\\\vaultcmd.exe</field>

```

```
<field name="win.eventData.commandLine" type="pcre2">list</field>
<description>Possible adversary activity - Attempt to list credentials via
$(win.eventData.Image) on $(win.system.computer) endpoint.</description>
<mitre>
  <id>T1003</id>
</mitre>
</rule>

</group>
```

 [Edit this page](#)



# Windows Agent ossec.conf

De standaard installatie route volgt het volgende `PATH` voor de `ossec`-configuratie: `C:\Program Files (x86)\ossec-agent`. De aanvullende, vereiste configuratie is nodig voor de usecases beschreven in hoofdstuk 4 uit het Technisch Ontwerp en te vinden in de bijlage, genaamd configuratiebestand Windows `ossec.conf`:

Herstart de agent op Windows na elke aanpassing op het `ossec.conf`-bestand met het volgende commando: `Restart-Service -Name Wazuh`

Hieronder vind je het `ossec.conf`-configuratiebestand voor de Wazuh Windows agent:

```
<!--
  Wazuh - Agent - Default configuration for Windows
  More info at: https://documentation.wazuh.com
  Mailing List: https://groups.google.com/forum/#!forum/wazuh
-->

<ossec_config>

  <client>
    <server>
      <address>127.0.0.1</address>
      <port>1514</port>
      <protocol>tcp</protocol>
    </server>
    <config-profile>windows, windows10</config-profile>
    <crypto_method>aes</crypto_method>
    <notify_time>10</notify_time>
    <time-reconnect>60</time-reconnect>
    <auto_restart>yes</auto_restart>
    <enrollment>
      <enabled>yes</enabled>
      <agent_name>GONGOELOE</agent_name>
    </enrollment>
  </client>

  <!-- Agent buffer options -->
  <client_buffer>
    <disabled>no</disabled>
    <queue_size>5000</queue_size>
    <events_per_second>500</events_per_second>
```

```
</client_buffer>

<localfile>
<location>Security</location>
<log_format>eventchannel</log_format>
<query>
    Event[System[EventID=4688]
        and EventData[Data[@Name='NewProcessName']='C:\\Windows\\System32\\WindowsPower
</query>
</localfile>

<localfile>
    <location>System</location>
    <log_format>eventchannel</log_format>
</localfile>

<localfile>
    <location>Microsoft-Windows-Windows Defender/Operational</location>
    <log_format>eventchannel</log_format>
</localfile>

<localfile>
    <location>active-response\\active-responses.log</location>
    <log_format>syslog</log_format>
</localfile>

<localfile>
    <location>Microsoft-Windows-Sysmon/Operational</location>
    <log_format>eventchannel</log_format>
</localfile>

<rootcheck>
    <disabled>no</disabled>
    <windows_apps>./shared/win_applications_rcl.txt</windows_apps>
    <windows_malware>./shared/win_malware_rcl.txt</windows_malware>
</rootcheck>

<sca>
    <enabled>yes</enabled>
    <scan_on_start>yes</scan_on_start>
    <interval>12h</interval>
    <skip_nfs>yes</skip_nfs>
</sca>

<syscheck>
    <disabled>no</disabled>
    <frequency>43200</frequency>

    <directories recursion_level="0" restrict="regedit.exe$|system.ini$|win.ini$">%W
```

```

<directories recursion_level="0"
restrict="at.exe$|attrib.exe$|cacls.exe$|cmd.exe$|eventcreate.exe$|ftp.exe$|lsass.exe$|
<directories recursion_level="0">%WINDIR%\SysNative\drivers\etc</directories>
<directories recursion_level="0" restrict="WMIC.exe$">%WINDIR%\SysNative\wbem</directories>
<directories recursion_level="0" restrict="powershell.exe$">%WINDIR%\SysNative\powershell</directories>
<directories recursion_level="0" restrict="winrm.vbs$">%WINDIR%\SysNative\winrm</directories>
<directories recursion_level="0"
restrict="at.exe$|attrib.exe$|cacls.exe$|cmd.exe$|eventcreate.exe$|ftp.exe$|lsass.exe$|
<directories recursion_level="0">%WINDIR%\System32\drivers\etc</directories>
<directories recursion_level="0" restrict="WMIC.exe$">%WINDIR%\System32\wbem</directories>
<directories recursion_level="0" restrict="powershell.exe$">%WINDIR%\System32\powershell</directories>
<directories recursion_level="0" restrict="winrm.vbs$">%WINDIR%\System32\winrm</directories>
<directories realtime="yes">%PROGRAMDATA%\Microsoft\Windows\Start Menu\Programs\
</directories>

<ignore>%PROGRAMDATA%\Microsoft\Windows\Start Menu\Programs\Startup\desktop
<ignore type="sregex">.log$|.htm$|.jpg$|.png$|.chm$|.pnf$|.evtx$</ignore>

<frequency>300</frequency>
<windows_registry arch="both">HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\
<windows_registry arch="both">HKEY_LOCAL_MACHINE\Software\Microsoft\Windows\CurrentVersion\

<registry_ignore>HKEY_LOCAL_MACHINE\Security\Policy\Secrets</registry_ignore>
<registry_ignore>HKEY_LOCAL_MACHINE\Security\SAM\Domains\Account\Users</registry_ignore>
<registry_ignore type="sregex">\\Enum$</registry_ignore>
<registry_ignore>HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\MpsSvc\
<registry_ignore>HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\MpsSvc\
<registry_ignore>HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\MpsSvc\
<registry_ignore>HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\MpsSvc\
<registry_ignore>HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\MpsSvc\
<registry_ignore>HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\MpsSvc\
<registry_ignore>HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\MpsSvc\
<registry_ignore>HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\PolicyAgent\
<registry_ignore>HKEY_LOCAL_MACHINE\System\CurrentControlSet\Services\ADOVMPF\

<windows_audit_interval>60</windows_audit_interval>
<process_priority>10</process_priority>
<max_eps>50</max_eps>

<synchronization>
  <enabled>yes</enabled>
  <interval>5m</interval>
  <max_eps>10</max_eps>
</synchronization>
</syscheck>

<wodle name="syscollector">
  <disabled>no</disabled>
  <interval>1h</interval>
  <scan_on_start>yes</scan_on_start>
  <hardware>yes</hardware>

```

```
<os>yes</os>
<network>yes</network>
<packages>yes</packages>
<ports all="no">yes</ports>
<processes>yes</processes>
<synchronization>
  <max_eps>10</max_eps>
</synchronization>
</wodle>

<wodle name="cis-cat">
  <disabled>yes</disabled>
  <timeout>1800</timeout>
  <interval>1d</interval>
  <scan-on-start>yes</scan-on-start>
  <java_path>\\server\jre\bin\java.exe</java_path>
  <ciscat_path>C:\cis-cat</ciscat_path>
</wodle>

<wodle name="osquery">
  <disabled>yes</disabled>
  <run_daemon>yes</run_daemon>
  <bin_path>C:\Program Files\osquery\osqueryd</bin_path>
  <log_path>C:\Program Files\osquery\log\osqueryd.results.log</log_path>
  <config_path>C:\Program Files\osquery\osquery.conf</config_path>
  <add_labels>yes</add_labels>
</wodle>

<wodle name="eventchannel">
  <enabled>yes</enabled>
  <read_interval>5</read_interval>
  <location>Security</location>
  <query>Event/System[EventID=1102]</query>
</wodle>

<active-response>
  <disabled>no</disabled>
  <ca_store>wpk_root.pem</ca_store>
  <ca_verification>yes</ca_verification>
</active-response>

<logging>
  <log_format>plain</log_format>
</logging>

</ossec_config>
```

