

# Handleiding – Het opzetten van een automatische security test in Azure

**Teamleden:**

Mucayit Orakci  
Ali Butt  
San Cheung  
Melissa Blokland  
Bas Pallada  
Laila Aouraghe

**Datum:** 18-06-2021

**Vak:** Project Information Security

**Versie:** 1.0

## Inhoudsopgave

1.	Inleiding .....	3
2.	Opzetten en verbinden.....	4
3.	Build self-hosted Agent pool .....	7
3.1.	Algemeen.....	7
3.2.	PAT token creëren .....	8
3.3.	Linux Agent Pool.....	9
4.	Configuraties in de agent omgeving.....	11
4.1.	Installeren Powershell op Linux server .....	11
4.2.	Docker installatie.....	12
5.	Security pipeline .....	13
5.1.	Pipeline run check .....	13
5.2.	Scan met Bash en Powershell.....	14
5.3.	Scan met OWASP ZAP Scanner.....	18
6.	Test resultaten.....	21
7.	YAML pipeline scripts .....	23
7.1.	YAML Optie 1.....	23
7.2.	YAML Optie 2.....	25
8.	Bijlage	

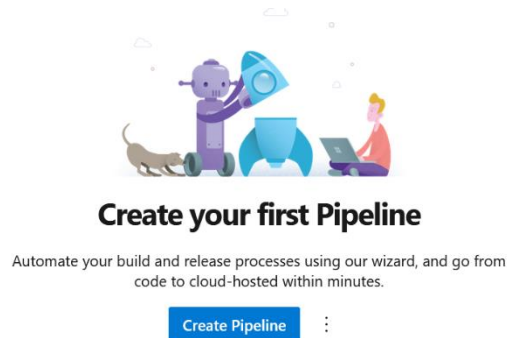
## 1. Inleiding

Deze handleiding is opgesteld om een stapsgewijze instructie te bieden voor het bouwen van een pipeline met een OWASP kwetsbaarheidsscanner. Voor de instructies wordt gebruik gemaakt van een Azure DevOps omgeving. In de handleiding worden twee manieren weergegeven van het implementeren van kwetsbaarheidsscanners, namelijk de scan met Bash en Powershell en de OWASP ZAP Scanner uit de store van Azure. Beide scripts maken gebruik van de OWASP top 10 om een applicatie op kwetsbaarheden te scannen. Het verschil tussen deze twee scripts zit meer in het publiceren van rapporten, de tijd waarin een scan wordt uitgevoerd en de hoeveelheid en niveau van handelingen voor installatie. Verder komen er in de handleiding zaken aan de orde als bijvoorbeeld het opzetten en verbinden van een pipeline, een self-hosted agent pool creëren, Docker installeren en het beschikbaar stellen van de rapporten van de security scan.

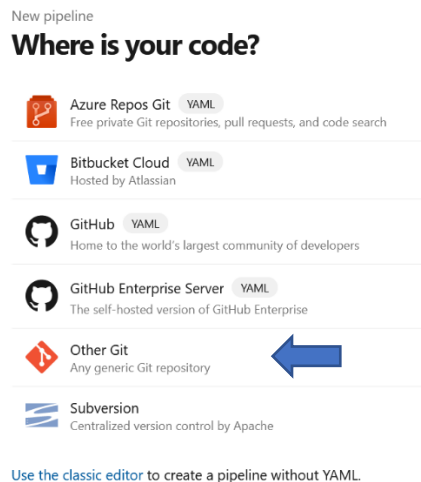
## 2. Opzetten en verbinden

In dit hoofdstuk worden instructies gegeven over hoe een pipeline opgezet kan worden en hoe deze verbonden kan worden met de repository.

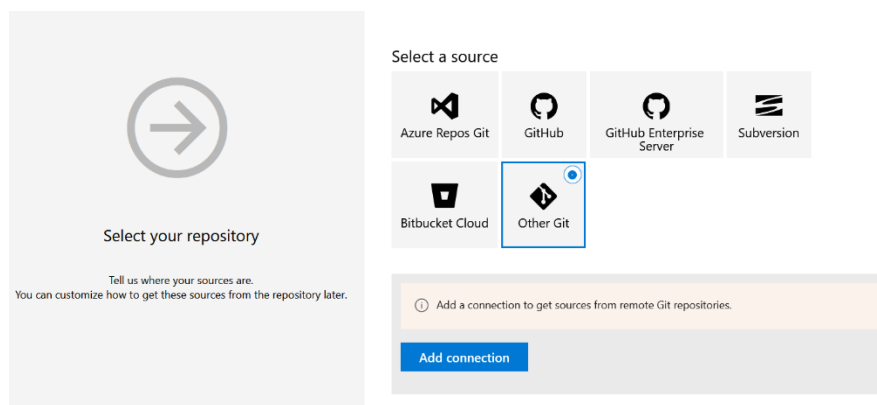
1. Om te beginnen is het noodzakelijk in Azure een project op te starten met daarbij eigen voorkeursinstellingen.
2. Zodra er een project is aangemaakt, klik in het linker menu op 'pipelines' en klik vervolgens 'Create pipeline'.



3. Dan vraagt het systeem waar de code zich bevindt. In dit voorbeeld gaan we de pipeline verbinden met Gitlab. Klik op de optie 'Other Git' om te verbinden met Gitlab.



Het volgende scherm wordt daarbij weergegeven:



4. Klik op 'Add connection' om een nieuwe verbinding te maken met een repository.

Add a generic Git service connection ✕

Connection name \*

connection-gitlab

Git repository URL \*

This setting is required.

User name

Password / Token key

OK Cancel

5. Bepaal een naam voor de connection en geef de Git repository URL op. Deze kan het beste behaald worden door in Gitlab naar het desbetreffende project te gaan en vervolgens te klikken op 'Clone with https'.

Clone with SSH

git@gitlab.fdmci.hva.nl:palladb

Clone with HTTPS

https://gitlab.fdmci.hva.nl/pal

Open in your IDE

Copy URL


6. Voor de 'User name' kan de gebruikersnaam van het Gitlab account opgegeven worden. De 'Token key' kan aangemaakt worden in Gitlab. Ga daarvoor naar user settings door uw avatar in de rechterbovenhoek te selecteren en klik op preferences in de drop-down. In het linker menu klik op access tokens.
7. Creëer een personal access token en gebruik de PAT key voor 'Password / Token key' in Azure. Klik vervolgens op 'OK'.

Name	Created	Last Used	Expires	Scopes	
pat	Jun 1, 2021	Never	In over 1 year	api, read_user, read_api, read_repository, write_repository, read_registry, write_registry	Revoke

8. Als de service connection is toegevoegd dient de 'Default branch for manual and scheduled builds' aangepast te worden. Deze staat automatisch op 'master', maar door een wijziging in

naamgebruik van Git moet deze gewijzigd worden naar 'main'. Als dit niet gedaan wordt, kan het systeem de repository hoogstwaarschijnlijk niet vinden. Klik vervolgens op continue.


Default branch for manual and scheduled builds

Default branch for manual and scheduled builds






9. In het volgende scherm kan er gestart worden met een 'Empty job'.

Select a template




Or start with an  Empty job


10. Dan wordt de pipeline weergegeven. De naam van de pipeline kan hier ook gewijzigd worden. Verder kunnen verschillende scripts worden gebouwd. Voordat we de pipeline kunnen runnen is het van belang om een agent pool beschikbaar te hebben. Deze gaan we in de volgende stappen creëren. Wel kunnen de instellingen alvast op default gezet worden bij 'Pipeline' en bij 'Agent job 1'. Deze moeten op default worden gezet, omdat we een self-hosted agent pool gaan opzetten. Sla de pipeline voor nu op.

 ... > Security-pipeline-CI

Tasks Variables Triggers Options Retention History |  Save & queue  Discard  Summary  Queue ... 



**Pipeline** Build pipeline ...


 Get sources  repository  main


Agent job 1  Run on agent +






Name \*




Agent pool \*  | [Pool information](#) | [Manage](#) 



Default 

Parameters 

Tasks Variables Triggers Options Retention History |  Save & queue  Discard  Summary  Queue ... 



**Pipeline** Build pipeline ...


 Get sources  repository  main

Agent job 1  Run on agent + 


Agent job 1

Agent selection ^


Agent pool  | [Pool information](#) | [Manage](#) 


Default 

**Hosted**

 Azure Pipelines

**Private**

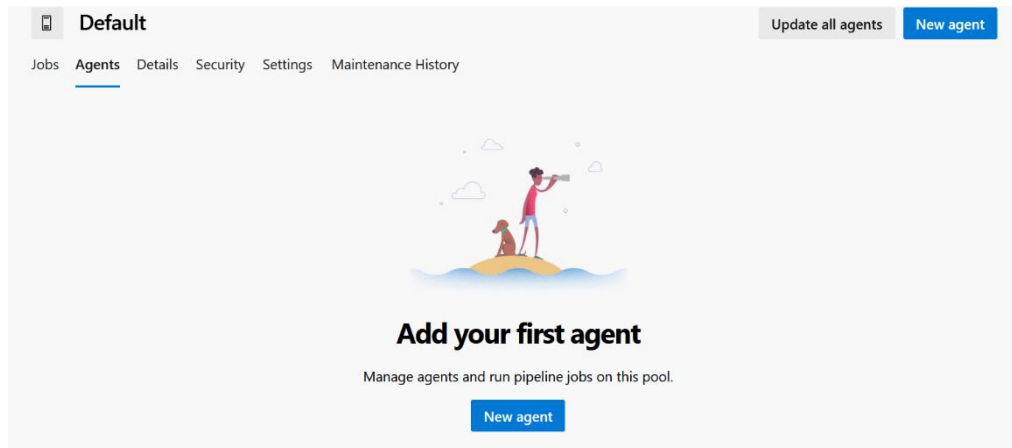
 <inherit from pipeline>

 Default

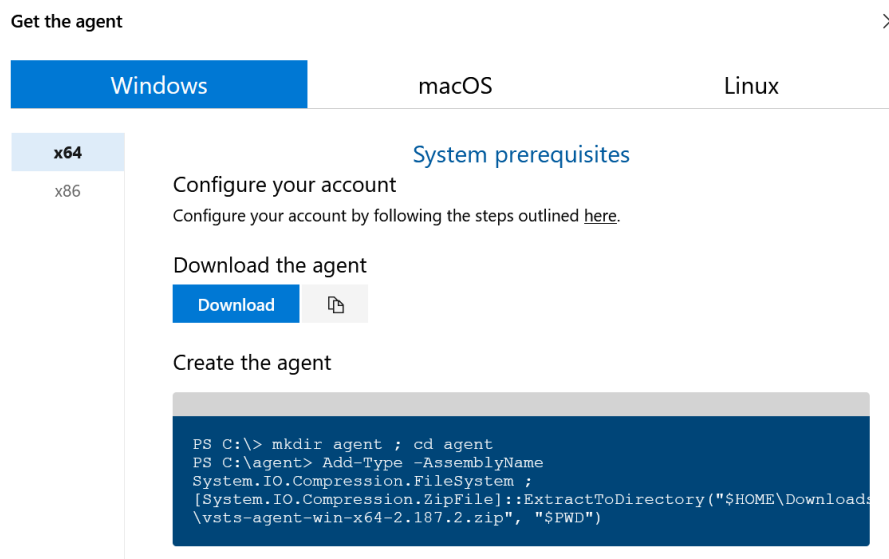
### 3. Build self-hosted Agent pool

#### 3.1. Algemeen

1. Als u op de main pagina van Azure bent, klik links onderin op 'Organization settings'. Scroll vervolgens naar beneden en klik op 'Agent pools'.
2. Klik dan op de optie 'Default'. En hierin gaat u naar 'Agents'.



3. Klik op 'New agent'. Het volgende scherm wordt weergegeven:



Er zijn drie opties om een agent te installeren. In deze handleiding worden instructies gegeven voor het installeren van een Linux agent. Voor dit project wordt aangeraden om een Linux agent aan te maken, omdat er security scripts gebouwd moeten worden die gebruik maken van Linux commands, zoals 'chmod'. Bij een Windows installatie zullen deze scripts hoogstwaarschijnlijk niet werken. Voor een gedetailleerde beschrijving zouden wij u willen verwijzen naar de technische documentatie.

### 3.2. PAT token creëren

1. Om een Agent Pool op te zetten moet er een PAT token gegenereerd worden. Navigeer naar de Azure DevOps hoofdpagina. Klik vervolgens rechts bovenin op 'User Settings' en klik op Personal Access Tokens in de dropdown.

### Personal Access Tokens

These can be used instead of a password for applications like Git or can be passed in the authorization header to access REST APIs

[+ New Token](#) [↶ Revoke](#) [✎ Edit](#) [↻ Regenerate](#)

Klik dan op 'New Token' en het volgende scherm komt in beeld:

### Create a new personal access token

×

Name

Organization

{Your organization} ▾

Expiration (UTC)

30 days ▾

6-7-2021

Scopes

Authorize the scope of access associated with this token

Scopes ☒ Full access ☐ Custom defined

Geef de PAT een naam en een expiration datum. Klik bij Scope 'Full access' aan. Als alles is ingevuld, klikt u op 'Create'.

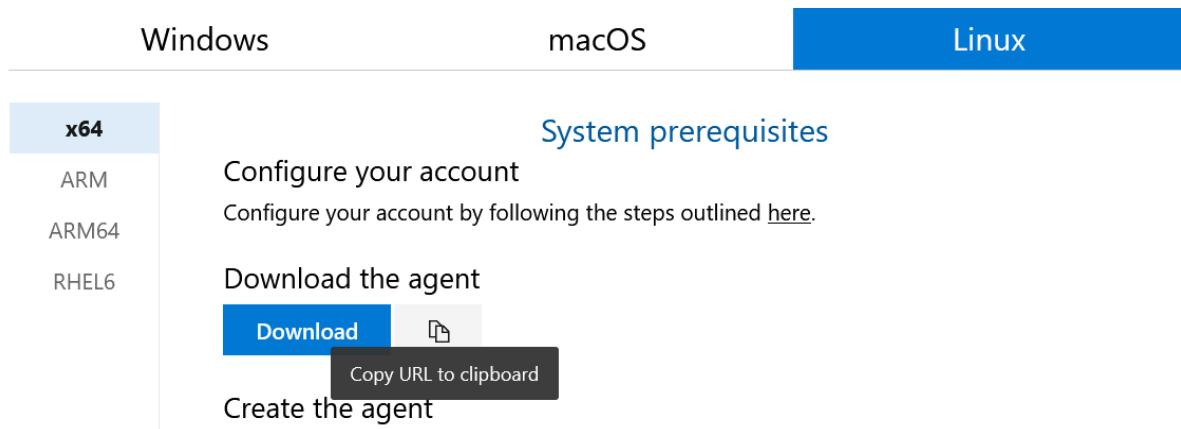
Vervolgens komt er een PAT token met de volgende melding. Als u in het vervolg geen gebruik wilt maken van deze token kunt u hem nu kopiëren en het venster sluiten. Als u hem in het vervolg wel nodig gaat hebben, dan wordt aangeraden de token ergens veilig op te slaan of te noteren, omdat deze hierna niet meer ingezien kan worden.

**Warning** - Make sure you copy the above token now. We don't store it and you will not be able to see it again.



### 3.3. Linux Agent Pool

1. Klik in het scherm voor 'New agent' op het tabblad 'Linux'. Als de agent pool geïnstalleerd moet worden op de computer waarmee ook op Azure is ingelogd, dan kan er gebruik worden gemaakt van de 'Download' knop, waarna het bestand op de computer in de download map geïnstalleerd wordt. In het geval dat de agent op een extern verbonden computer of server geïnstalleerd moet worden, dan klikt u op 'Copy URL to clipboard'. In deze instructie gaan we verder met het installeren van een agent op een extern systeem en dus via de commandline. Dit betekent ook dat we een iets andere aanpak hanteren van het installeren dan dat Azure geeft, omdat dit een gebruiksvriendelijkere aanpak heeft.



2. Maak op de computer waar de agent geïnstalleerd moet worden een nieuwe map, genaamd 'myagent' en navigeer vervolgens naar die map.
3. Om de agent te downloaden geef het volgende command op:

```
wget {URL copied to clipboard}
```

4. In deze stap gaan we de agent creëren. Omdat de gedownloade agent nu niet in de map downloads staat, voeren we de volgende opdracht uit:

```
tar zxvf vsts-agent-linux-x64-2.187.2.tar.gz
```

5. Dan gaan we de agent configureren. Hierbij is het belangrijk dat we nog steeds in de map 'myagent' zitten. We kunnen dan de volgende opdracht uitvoeren:

```
./config.sh
```

6. Het systeem vraagt naar een aantal zaken. Hierna volgen de vragen met de instructies over welk antwoord u moet opgeven:

```
Enter (Y/N) Accept the Team Explorer Everywhere license agreement now? (press enter for N) > Enter
```

```
Enter server URL > https://dev.azure.com/{your_organization}
```

```
Enter authentication type (press enter for PAT) > Enter
```

```
Enter personal access token > { Your PAT token } (zie 'PAT token creëren')  
>> Register Agent:
```

Enter agent pool (press enter for default) > **Enter**

Enter agent name (press enter for azure01) > **{Typ a name else Enter}**

Enter work folder (press enter for \_work) > **Enter**

7. De settings zijn gesaved. Iedere keer als de pipeline opdrachten uit moet voeren, moet de agent ook draaien. Om de agent te runnen navigeert u naar de map 'myagent' en geeft u het volgende commando op:

`./run.sh`

## 4. Configuraties in de agent omgeving

In dit hoofdstuk worden zaken behandeld die, afhankelijk van uw omgeving en methoden, nodig zijn om de security scripts juist te kunnen uitvoeren in de pipeline. Hiervoor is beschreven hoe een agent pool opgezet kan worden, maar hierbij moeten nog wat extra programma's worden geïnstalleerd of gedraaid. In dit hoofdstuk bespreken we daarom het installeren van Powershell op een Linux server en het draaien van Docker op de server. Deze configuraties vinden allen plaats in de omgeving van de agent waar de pipeline op draait. Mocht dit uw huidige computer zijn, dan zullen deze configuraties op uw computer moeten plaatsvinden, maar draait de agent op een externe pc of server, dan zullen deze instructies ook op die externe omgeving uitgevoerd moeten worden.

### 4.1. Installeren Powershell op Linux server

Voor het uitvoeren van security scripts via methode 1 (later in de handleiding) is het noodzakelijk dat de Linux server de mogelijkheid heeft om Powershell te kunnen runnen, in verband met het uitvoeren van een Powershell script. Deze is in de meeste gevallen niet standaard geïnstalleerd op Linux. Het is mogelijk dit wel te hebben, mits deze is geïnstalleerd voor de juiste versie. Hieronder staat een lijst beschreven met de Powershell versie en de ondersteunende platformen:

Officieel ondersteunde platform releases voor Power shell 7,1:

- Ubuntu 16.04/18.04/20.04 (inclusief ARM64)
- Ubuntu 19,10 (via snap package)
- Debian 9/10
- CentOS en RHEL 7/8
- Fedora 30
- Alpiene 3.11 + (inclusief ARM64)

Officieel ondersteunde platform releases voor Power shell 7,0:

- Ubuntu 16.04
- Ubuntu 18,04 en 20,04
- Debian 8
- Debian 9
- Debian 10
- Alpine 3,9 en 3,10
- CentOS 7
- Red Hat Enterprise Linux (RHEL) 7
- Fedora 28
- Fedora 29
- Fedora 30
- openSUSE 42,3
- openSUSE Schrikkel 15

Om Powershell te installeren op Linux via de commandline kunt u gebruik maken van het volgende commando:

```
sudo apt-get install -y powershell.
```

## 4.2. Docker installatie

In deze paragraaf worden instructies gegeven over het installeren van Docker. Voor deze installatie is er uitgegaan van Ubuntu. Op <https://docs.docker.com/engine/install/> zijn installatiehandleidingen te vinden van andere versies.

1. Allereerst moeten er een aantal packages gedownload worden die apt packages laten installeren over HTTPS:

```
sudo apt install apt-transport-https ca-certificates curl software-properties-common
```

2. Daarna moet de GPG key van de officiële Docker repository toegevoegd worden aan het systeem:

```
curl -fsSL https://download.docker.com/linux/ubuntu/gpg | sudo apt-key add -
```

3. De Docker repository toevoegen aan de APT bronnen (andere versies voor ARM zijn te vinden op:

```
sudo add-apt-repository "deb [arch=amd64]  
https://download.docker.com/linux/ubuntu focal stable"
```

4. Update de package database met de nieuwe docker packages:

```
sudo apt update
```

5. Zorg dat apt de docker engine installeert vanuit de Docker repo en niet vanuit de standaard Ubuntu repo:

```
apt-cache policy docker-ce
```

6. Installeer Docker:

```
sudo apt install docker-ce
```

7. Controleer dat Docker geïnstalleerd is en draait:

```
Sudo systemctl status docker
```

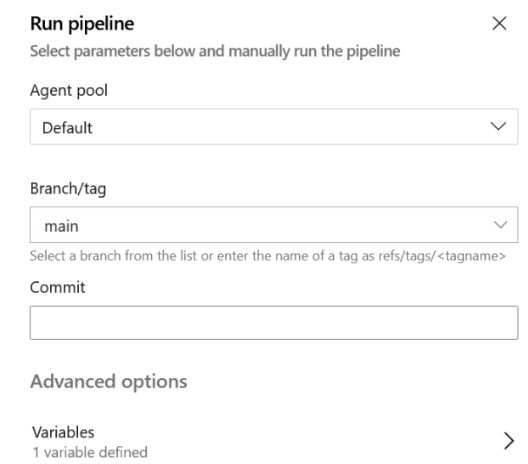
## 5. Security pipeline

In dit hoofdstuk wordt uitgelegd hoe de security scripts kunnen worden geïmplementeerd in de pipeline. Daarbij zullen ook scripts worden toegevoegd die ervoor zorgen dat de resultaten van de scans zichtbaar worden. De scripts kunnen gebouwd worden op meerdere manieren. Hieronder bespreken we er twee, namelijk met 1) Bash en Powershell en met 2) OWASP ZAP Scanner uit de Azure market place.

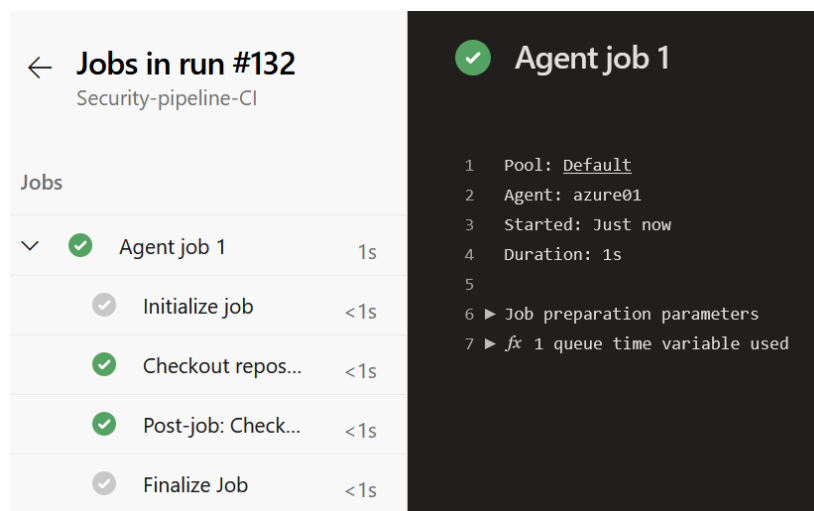
### 5.1. Pipeline run check

Als het de eerste keer is dat u de pipeline opstart na het opzetten van een agent, kunnen deze stappen worden doorlopen. Deze stappen zijn om te controleren of de agent pool werkt met de pipeline. Indien dit voor u niet nodig is, kan deze paragraaf worden overgeslagen.

1. Ga naar uw pipeline en run vervolgens de pipeline. Er komt een scherm, waarop u nog bepaalde instellingen kunt opgeven. Controleer hier ook of de Agent pool op 'Default' staat. De self-hosted agent pool die we hiervoor hebben gebouwd is een onderdeel van Default en pakt hij daarmee automatisch als deze online staat. Klik dan op 'Run'.



2. Als het resultaat van 'Agent job 1' op groen eindigt en de instellingen op de volgende manier weergeeft, dan is de agent juist geïnstalleerd.



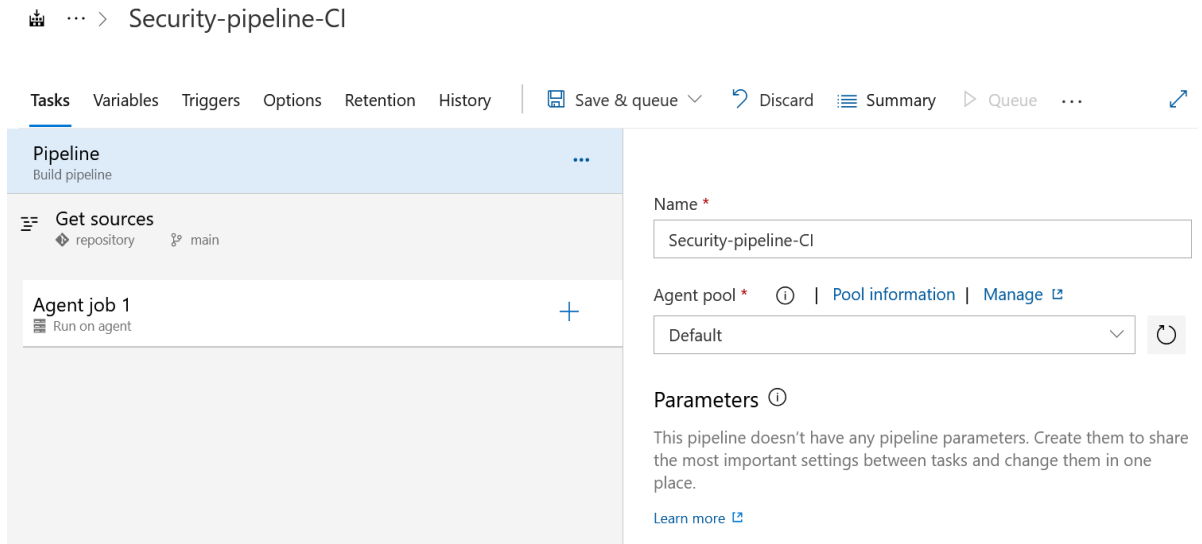
Jobs in run #132		
Security-pipeline-CI		
Jobs		
✓	Agent job 1	1s
✓	Initialize job	<1s
✓	Checkout repos...	<1s
✓	Post-job: Check...	<1s
✓	Finalize Job	<1s

✓ Agent job 1

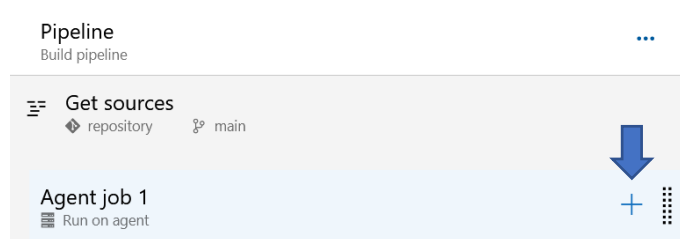
```
1 Pool: Default
2 Agent: azure01
3 Started: Just now
4 Duration: 1s
5
6 ► Job preparation parameters
7 ► fx 1 queue time variable used
```

## 5.2. Scan met Bash en Powershell

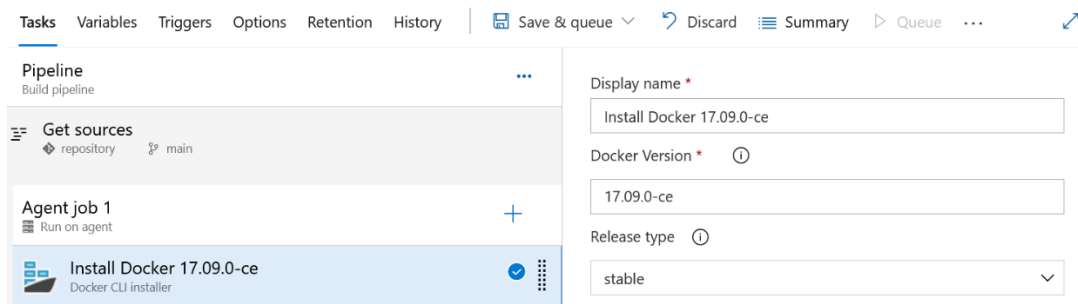
1. Om een pipeline te bewerken gaat u naar Pipelines, klikt u uw pipeline aan en vervolgens op 'Edit'. U krijgt het volgende scherm:



2. Klik vervolgens op 'Add task to Agent Job 1' door op de plus te klikken.



3. Zoekt u in de lijst naar 'Docker CLI installer' en voegt u deze toe aan de pipeline. De volgende instellingen kunnen daarbij gegeven worden (vaak staan deze standaard al goed):



Docker version: 17.09.0-ce  
Release type: stable

4. Voeg vervolgens nog een task aan de Agent job, genaamd 'Bash' en geef daarbij de volgende instellingen op:

The screenshot shows the 'OWASP-Scan' task configuration in the Azure Pipelines interface. The task is part of 'Agent job 1' and is of type 'Bash'. The configuration panel on the right shows the following settings:

- Task version: 3.\*
- Display name: OWASP-Scan
- Type: Inline (selected)
- Script: 

```
chmod -R 777 ./  
  
docker run --rm -v $(pwd)/zap/wrk/:rw -t owasp/zap2docker-stable zap-full-scan.py -t {scan-website} -g gen.conf -x OWASP-ZAP-Report.xml -r scan-report.html  
  
true
```

Task version: 3  
Type: Inline  
Script:

```
chmod -R 777 ./  
  
docker run --rm -v $(pwd)/zap/wrk/:rw -t owasp/zap2docker-stable zap-full-scan.py -t {scan-website} -g gen.conf -x OWASP-ZAP-Report.xml -r scan-report.html  
  
true
```

- Met scan-website wordt bedoeld, de website die gescand moet worden zonder haakjes!  
Bijvoorbeeld: <https://www.yourdomainhere.com>
- Zorg dat u wel toestemming heeft van de eigenaar om de website te scannen! Dit is strafbaar als u geen toestemming heeft.

5. Dit is een tussenstap om het script uit te kunnen voeren die we hierna gaan opzetten.

Navigeer naar onderstaande link om het bestand te downloaden en noem het OWASPToNUnit3.xslt:

[https://dev.azure.com/francislacroix/\\_git/CodeShare?path=%2FOWASPB1og%2FOWASPToNUnit3.xslt](https://dev.azure.com/francislacroix/_git/CodeShare?path=%2FOWASPB1og%2FOWASPToNUnit3.xslt)

Push dit bestand vervolgens met deze naam in de repository. Als deze repository niet op een systeem of computer van de agent staat, dan moet dit alsnog gedaan worden, zodat het xslt-bestand gevonden kan worden. Dat kan met de volgende stappen:

- Op de agent pool, navigeer naar de map, waar de map 'myagent' in staat.

```
prosecure@azure01:~/instructie$ ls  
myagent
```

- Clone de repository in deze map:

```
prosecure@azure01:~/instructie$ ls  
myagent repository
```

- Zorg dat het bestand 'OWASPToNUnit3.xslt' in de map van de repository zit. Zo niet, dan kan hij hier alsnog aan toegevoegd worden.
- Sla het pad op waar het bestand 'OWASPToNUnit3.xslt' op de agent gevonden kan worden. Bijvoorbeeld:  
"/home/prosecure/repository/OWASPToNUnit3.xslt"

6. Terug in de pipeline, voeg nog een task aan de Agent job, genaamd 'PowerShell'. Geef de volgende instellingen op:

The screenshot shows the Azure Pipelines configuration page for a pipeline named 'Build pipeline'. The 'Agent job 1' section contains four tasks: 'Get sources' (repository), 'Install Docker 17.09.0-ce' (Docker CLI installer), 'OWASP-Scan' (Bash), and 'PowerShell Script' (PowerShell). The 'PowerShell Script' task is selected, and its configuration is shown on the right. The 'Display name' is 'PowerShell Script', the 'Type' is 'Inline', and the 'Script' is a PowerShell script that uses XSLT to transform an OWASP ZAP report.

Tasks Variables Triggers Options Retention History | Save & queue Discard Summary Queue ...

Pipeline  
Build pipeline

Get sources  
repository main

Agent job 1  
Run on agent

Install Docker 17.09.0-ce  
Docker CLI installer

OWASP-Scan  
Bash

PowerShell Script  
PowerShell

Display name \*  
PowerShell Script

Type ⓘ  
☐ File Path ☒ Inline

Script \* ⓘ

```
$XslPath = "/home/prosecure/pro-secure-  
test/OWASPToNUnit3.xslt"  
$XmlInputPath =  
"$($Env:SYSTEM_DEFAULTWORKINGDIRECTORY)/OWASP-ZAP-  
Report.xml"  
$XmlOutputPath =  
"$($Env:SYSTEM_DEFAULTWORKINGDIRECTORY)/Converted-  
OWASP-ZAP-Report.xml"  
$XslTransform = New-Object  
System.Xml.Xsl.XslCompiledTransform  
$XslTransform.Load($XslPath)
```

Task version: 2

Type: Inline

Script:

```
$XslPath = (path op agent naar OWASPToNUnit3.xslt in repository, zie stap 5 hiervoor)  
$XmlInputPath = "$($Env:SYSTEM_DEFAULTWORKINGDIRECTORY) /OWASP-  
ZAP-Report.xml"  
$XmlOutputPath =  
"$($Env:SYSTEM_DEFAULTWORKINGDIRECTORY) /Converted-OWASP-ZAP-  
Report.xml"  
$XslTransform = New-Object System.Xml.Xsl.XslCompiledTransform  
$XslTransform.Load($XslPath)  
$XslTransform.Transform($XmlInputPath, $XmlOutputPath)
```



7. Deze stap is om de resultaten van de scan in Azure te bekijken. Voeg vervolgens nog een task toe aan de Agent job, genaamd 'Publish Test Results'. De instellingen zijn als volgt:

The screenshot shows the configuration for the 'Publish Test Results' task in an Azure DevOps pipeline. The task is added to 'Agent job 1'. The configuration panel on the right shows the following settings: Task version is set to '2.\*'; Display name is 'Publish Test Results'; Test result format is set to 'NUnit'; Test results files is set to 'Converted-OWASP-ZAP-Report.xml'; and Search folder is empty.

Task version: 2  
Test result format: NUnit  
Test results files: Converted-OWASP-ZAP-Report.xml  
Search folder: \$(System.DefaultWorkingDirectory) -> Afhankelijk van de instellingen van de agent, is dit /myagent/\_work

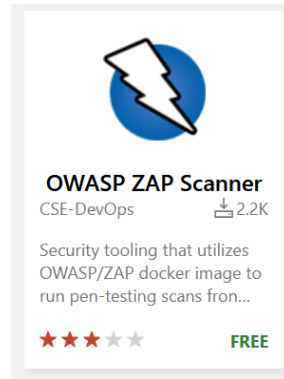
8. Deze taak is om een html report te kunnen downloaden meteen na de scan. Voeg vervolgens nog een task aan de Agent job, genaamd 'Publish build artifacts'. Geef de volgende instellingen op:

The screenshot shows the configuration for the 'Publish Artifact: owasp-result' task in an Azure DevOps pipeline. The task is added to 'Agent job 1'. The configuration panel on the right shows the following settings: Task version is set to '1.\*'; Display name is 'Publish Artifact: owasp-result'; Path to publish is set to '\$(Build.SourcesDirectory)/scan-report.html'; Artifact name is set to 'owasp-result'; and Artifact publish location is set to 'Azure Pipelines'.

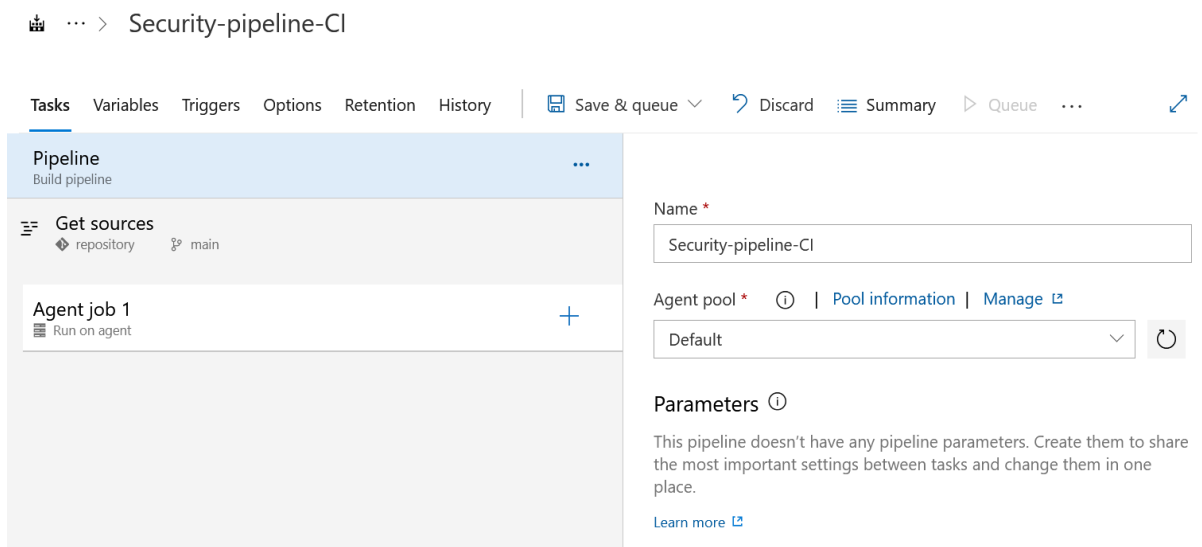
Path to publish: \$(Build.SourcesDirectory)/scan-report.html  
Artifact publish location: Azure pipelines

### 5.3. Scan met OWASP ZAP Scanner

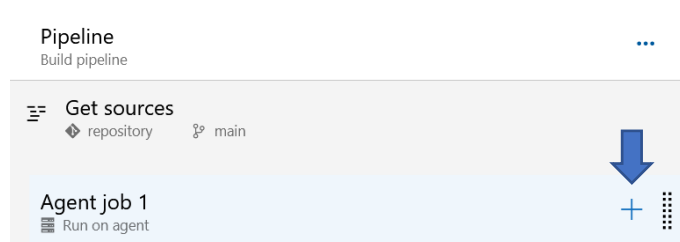
1. In deze stap gaan we het OWASP script installeren via de Azure Market place, waar naar genavigeerd kan worden via het menu rechts bovenin.
2. Vul in de zoekbalk 'OWASP ZAP scanner' en installeer de volgende:



3. Om de pipeline te bewerken gaat u naar Pipelines, klikt u uw pipeline aan en vervolgens op 'Edit'. Het volgende scherm wordt weergegeven:



4. Klik vervolgens op 'Add task to Agent Job 1' door op de plus te klikken.



5. Zoekt u in de lijst naar 'Docker CLI installer' en voeg deze toe aan de pipeline. De volgende instellingen kunnen daarbij gegeven worden (vaak staan deze standaard al goed):

Tasks Variables Triggers Options Retention History | Save & queue Discard Summary Queue ...

**Pipeline**  
Build pipeline

**Get sources**  
repository main

**Agent job 1**  
Run on agent

**Install Docker 17.09.0-ce**  
Docker CLI installer

**Display name \***  
Install Docker 17.09.0-ce

**Docker Version \*** ⓘ  
17.09.0-ce

**Release type** ⓘ  
stable

Docker version: 17.09.0-ce  
Release type: stable

6. Voeg vervolgens de geïnstalleerde OWASP ZAP Scanner toe aan de pipeline en geef daarbij de volgende instellingen op:

Tasks Variables Triggers Options Retention History | Save & queue Discard Summary Queue ...

**Pipeline**  
Build pipeline

**Get sources**  
repository main

**Agent job 1**  
Run on agent

**Install Docker 17.09.0-ce**  
Docker CLI installer

**ZAP Scanner**  
OWASP Zap Scanner

**Failure Threshold** ⓘ  
1000

**Scan Type** ⓘ  
☒ Targeted Scan ☐ Scan on agent

**Root URL to begin crawling \*** ⓘ  
https://www.yourdomainhere.com

☐ Provide Context File ⓘ

**Scan Port** ⓘ  
443

Aggressive Scan Mode: True  
Failure Threshold: 1000  
Scan Type: Targeted Scan  
Root URL to begin crawling: {scan-website} e.g. <https://www.yourdomainhere.com>  
Scan Port 443

7. In de volgende stap wordt een script gebouwd die na iedere scan een script genereert. Voeg nog een task aan de Agent job, genaamd 'Publish build artifacts' en geef daarbij de volgende instellingen op:

Tasks
Variables
Triggers
Options
Retention
History
Save & queue
Discard
Summary
Queue
...

**Pipeline**  
Build pipeline

**Get sources**  
repository
main

**Agent job 1**  
Run on agent

**Install Docker 17.09.0-ce**  
Docker CLI installer

**ZAP Scanner**  
OWASP Zap Scanner

**Publish Artifact: owasp-report**  
Publish build artifacts

Display name \*  
Publish Artifact: owasp-report

Path to publish \* ⓘ  
\$(Build.SourcesDirectory)/owaspzap

Artifact name \* ⓘ  
owasp-report

Artifact publish location \* ⓘ  
Azure Pipelines

Advanced

Path to publish: \$(Build.SourcesDirectory)/owaspzap

Artifact publish location: Azure pipelines

## 6. Test resultaten

De resultaten van de scans zijn op meerdere plekken terug te vinden. Voor de scan met Bash en Powershell kunnen de testresultaten op drie manieren worden teruggevonden. Bij de OWASP ZAP Scanner kunnen de test resultaten alleen met optie 1 (hieronder weergegeven) bekeken worden. Het zou overigens ook geïmplementeerd kunnen worden met het implementeren van de bijbehorende tasks en opgeven van andere paden.


Security scan met Bash en Powershell: optie 1, 2, 3

Security scan met OWASP ZAP Scanner: optie 1

1. Nadat de test klaar is kan er geklikt worden op '1 artifact produced'. En selecteer vervolgens 'report.html'.

The screenshot shows the Azure DevOps interface. On the left, under 'Jobs in run #136', a list of jobs is shown, including 'Agent job 1' which is completed. In the center, the 'Agent job 1' console output is visible, showing steps like 'Job preparation parameters', 'Job live console data', and '1 artifact produced'. On the right, the 'Artifacts' section is open, showing a list of published artifacts: 'owasp-report', 'report.html', and 'report.json'.

Het html-bestand 'report.html' geeft een OWASP scan report dat er ongeveer als volgt uit ziet:

ZAP Scanning Report

Summary of Alerts

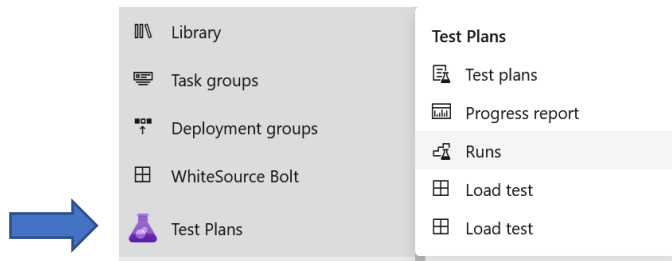
Generated on Sun, 6 Jun 2021 18:43:07

Risk Level	Number of Alerts
High	2
Medium	5
Low	4
Informational	4

Alerts

Name	Risk Level	Number of Instances
Cloud Metadata Potentially Exposed	High	1
PII Disclosure	High	2
.env Information Leak	Medium	3
Apache Range Header DoS (CVE-2011-3192)	Medium	14
Content Security Policy (CSP) Header Not Set	Medium	4
Trace.axd Information Leak	Medium	3
X-Frame-Options Header Not Set	Medium	4
Incomplete or No Cache-control and Pragma HTTP Header Set	Low	5
Server Leaks Version Information via "Server" HTTP Response Header Field	Low	12
Strict-Transport-Security Header Not Set	Low	12
X-Content-Type-Options Header Missing	Low	12
Information Disclosure - Suspicious Comments	Informational	5
Modern Web Application	Informational	6
Timestamp Disclosure - Unix	Informational	6
User Agent Fuzzer	Informational	14

2. De tweede optie is door in het linker menu 'Test plans' te selecteren en vervolgens te navigeren naar 'Runs'.



Daarbij worden de resultaten als volgt weergegeven:

Run 4 - NUnit\_TestResults\_134

Run summary **Test results** Filter

🔄 | 🐛 Create bug | ✎ Update analysis

Outcome	Test Case Title
❌ Failed	Cloud Metadata Potentially Exposed
❌ Failed	Information Disclosure - Suspicious Comments
❌ Failed	Modern Web Application
❌ Failed	Timestamp Disclosure - Unix
❌ Failed	X-Frame-Options Header Not Set
❌ Failed	Content Security Policy (CSP) Header Not Set
❌ Failed	Apache Range Header DoS (CVE-2011-3192)
❌ Failed	X-Content-Type-Options Header Missing
❌ Failed	.env Information Leak
❌ Failed	PII Disclosure
❌ Failed	Server Leaks Version Information via "Server" HTTP Response Header Fi...

Run 4 - NUnit\_TestResults\_134

Run summary **Test results** Filter

🔄 | ✎ Update comment

**Summary**

⚠ Completed 2 hours ago, Ran for 187 milliseconds

Run type	Automated
Owner	Security-pipeline
Build Service (mb5914)	
Tested build	134
Release	not available
Release Stage	not available
Build platform	Win32NT
Build flavor	not available
Test settings	Default
MTM lab environment	not available

**Comments**

No comments

**Outcome**

15

❌ Failed

Outcome by priority

3. De laatste optie is in de build van de pipeline. Daar kan op tests geklikt worden en worden er onder andere de kwetsbaarheden gegeven met een summary.

🟢 #137 on Security-pipeline-CI [Run new](#)

ⓘ This build will be retained forever by 9 (Pipeline) [View retention leases](#)

**Summary** Tests

Manually run by mb5914

Repository and version	Time started and elapsed	Related	Tests and coverage
📁 repository	📅 Today at 20:53	📦 0 work items	🟢 0% passed
📁 main	🕒 3m 18s	📄 1 published	<a href="#">Setup code coverage</a>

## 7. YAML pipeline scripts

In dit hoofdstuk wordt de YAML van beide opties voor security pipelines weergegeven. Wij hebben geen gebruik gemaakt van de optie om het zelf in één YAML file te typen, omdat wij tasks konden toevoegen aan de pipeline. Deze vertalen de tasks automatisch naar een YAML. Dit komt, omdat we gebruik maken van 'other Git', namelijk Gitlab. Mochten wij of u over willen gaan naar Github, dan is het handig om daarvoor een YAML script te hebben. U heeft dan de mogelijkheid het script in de YAML van de pipeline te plakken en de pipeline werkt op dezelfde manier als met tasks. Uiteraard zou u ongetwijfeld een aantal wijzigingen moeten aanbrengen, omdat u andere omgevingen, paden en benamingen gebruikt dan in deze instructie. De twee volledige YAML files zijn ook toegevoegd aan de bijlage van dit document.

### 7.1. YAML Optie 1

De volgende YAML wordt gebruikt voor optie 1 'Scan met Bash en Powershell':

```
-----
pool:
  name: Default
  demands: sudo

steps:
- task: DockerInstaller@0
  displayName: 'Install Docker 17.09.0-ce'

- bash: |
  chmod -R 777 ./

  docker run --rm -v $(pwd):/zap/wrk/:rw -t owasp/zap2docker-stable
  zap-full-scan.py -t https://clerqbot.com/ -g gen.conf -x OWASP-ZAP-
  Report.xml -r scan-report.html

  true

  displayName: 'OWASP-Scan'

- powershell: |
  $XslPath = "/home/prosecure/pro-secure-test/OWASPToNUnit3.xslt"

  $XmlInputPath = "$($Env:SYSTEM_DEFAULTWORKINGDIRECTORY)/OWASP-
  ZAP-Report.xml"

  $XmlOutputPath =
  "$($Env:SYSTEM_DEFAULTWORKINGDIRECTORY)/Converted-OWASP-ZAP-
  Report.xml"
```

```
$XslTransform = New-Object System.Xml.Xsl.XslCompiledTransform
$XslTransform.Load($XslPath)
$XslTransform.Transform($XmlInputPath, $XmlOutputPath)

displayName: 'Convert-owasp-report'

- task: PublishTestResults@2
  displayName: 'Publish Test Results'
  inputs:
    testResultsFormat: NUnit
    testResultsFiles: 'Converted-OWASP-ZAP-Report.xml'

- task: PublishBuildArtifacts@1
  displayName: 'Publish Artifact: owasp-report'
  inputs:
    PathToPublish: '$(Build.SourcesDirectory)/scan-report.html'
    ArtifactName: 'owasp-report'
```

---



## 7.2. YAML Optie 2

De volgende YAML wordt gebruikt voor optie 2 'Scan met OWASP Zap scan':

---

```
pool:
  name: Default
steps:
- task: DockerInstaller@0
  displayName: 'Install Docker 17.09.0-ce'

- task: CSE-DevOps.zap-scanner.custom-build-release-task.owaspzap@1
  displayName: 'ZAP Scanner'
  inputs:
    aggressivemode: true
    threshold: 1000
    scantype: targetedScan
    url: 'https://clerqbot.com/'
    port: 443

- task: PublishBuildArtifacts@1
  displayName: 'Publish Artifact: owasp-report'
  inputs:
    PathtoPublish: '$(Build.SourcesDirectory)/owaspzap'
    ArtifactName: 'owasp-report'
```

---