

Comment affiner les règles Wazuh pour éviter les faux positifs

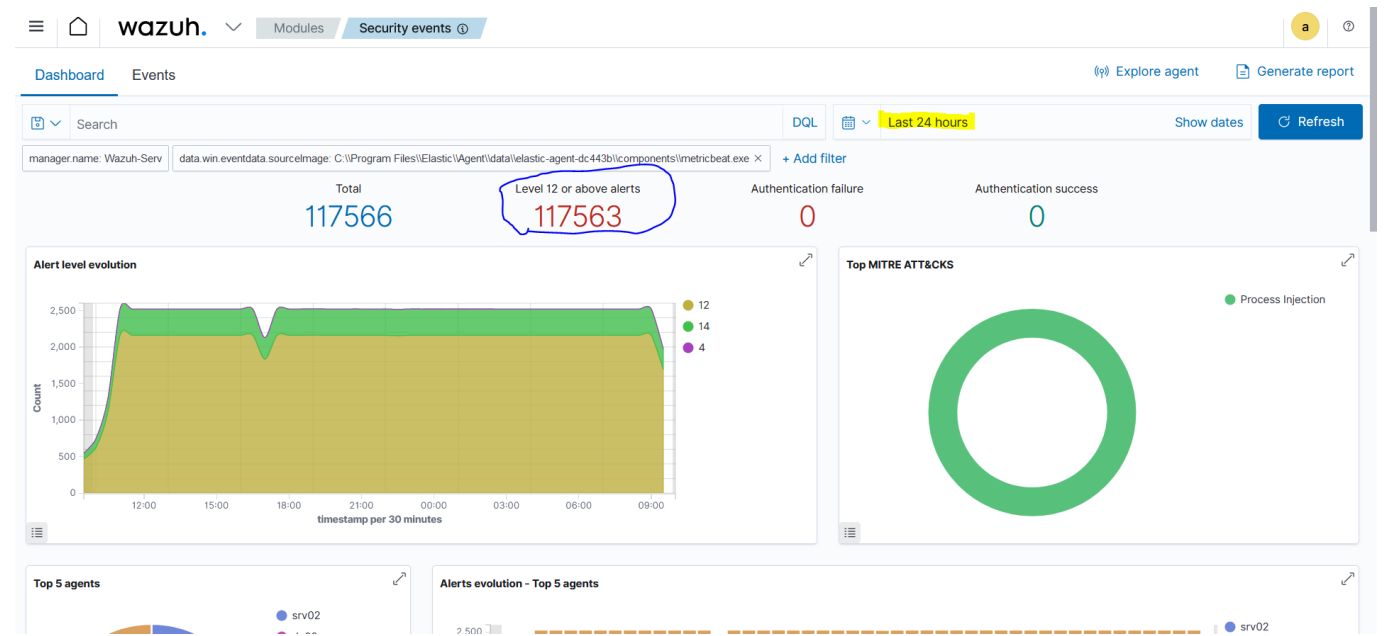
Introduction :

Après avoir installé le SIEM Elastic et déployé les agents Beats sur nos machines Windows, Wazuh a commencé à lever énormément d'alertes de niveau 12 (donc critique). Cependant, en regardant les détails de ces alertes, on remarque que c'est l'agent d'Elastic qui les provoquent :

>	Nov 29, 2023 @ 09:03:02.433	004	dc02	T1055	Defense Evasion, Privilege Escalation	Explorer process was accessed by C:\Program Files\Elastic\Agent\data\elastic-agent-dc443b\components\metricbeat.exe, possible process injection	12	92910
>	Nov 29, 2023 @ 09:03:02.416	004	dc02	T1055	Defense Evasion, Privilege Escalation	Explorer process was accessed by C:\Program Files\Elastic\Agent\data\elastic-agent-dc443b\components\metricbeat.exe, possible process injection	12	92910
>	Nov 29, 2023 @ 09:03:02.134	004	dc02	T1055	Defense Evasion, Privilege Escalation	Windows Remote Desktop utility process was accessed by C:\Program Files\Elastic\Agent\data\elastic-agent-dc443b\components\metricbeat.exe, possible process injection	14	92920
>	Nov 29, 2023 @ 09:03:02.134	004	dc02	T1055	Defense Evasion, Privilege Escalation	Windows Remote Desktop utility process was accessed by C:\Program Files\Elastic\Agent\data\elastic-agent-dc443b\components\metricbeat.exe, possible process injection	14	92920
>	Nov 29, 2023 @ 09:03:01.010	002	srv02	T1055	Defense Evasion, Privilege Escalation	Explorer process was accessed by C:\Program Files\Elastic\Agent\data\elastic-agent-dc443b\components\metricbeat.exe, possible process injection	12	92910
>	Nov 29, 2023 @ 09:03:01.009	002	srv02	T1055	Defense Evasion, Privilege Escalation	Explorer process was accessed by C:\Program Files\Elastic\Agent\data\elastic-agent-dc443b\components\metricbeat.exe, possible process injection	12	92910
>	Nov 29, 2023 @ 09:03:00.538	005	dc01	T1055	Defense Evasion, Privilege Escalation	Explorer process was accessed by C:\Program Files\Elastic\Agent\data\elastic-agent-dc443b\components\metricbeat.exe, possible process injection	12	92910

Pour voir l'ampleur des faux positifs, on peut faire une query poussée avec OpenSearch comme ceci :

```
{
  "query": {
    "match_phrase": {
      "data.win.eventdata.sourceImage": "C:\\\\Program
Files\\\\Elastic\\\\Agent\\\\data\\\\elastic-agent-
dc443b\\\\components\\\\metricbeat.exe"
    }
  }
}
```



En moins de 24 heures, on a près de 118 000 alertes critiques uniquement causées par l'agent metricbeat.exe. Cela empêche donc de pouvoir visualiser correctement les vraies alertes que Wazuh doit relever. Il faut donc rajouter une exception pour arrêter de loguer l'agent.

Résolution :

On commence par regarder quelle ID de règle génère les alertes sur le côté droit de notre Dashboard de sécurité :

Tactic(s)	Description	Level	Rule ID
Defense Evasion, Privilege Escalation	Explorer process was accessed by C:\\Program Files\\Elastic\\Agent\\data\\elastic-agent-dc443b\\components\\metricbeat.exe, possible process injection	12	92910
Defense Evasion, Privilege Escalation	Explorer process was accessed by C:\\Program Files\\Elastic\\Agent\\data\\elastic-agent-dc443b\\components\\metricbeat.exe, possible process injection	12	92910
Defense Evasion, Privilege Escalation	Windows Remote Dektop utility process was accessed by C:\\Program Files\\Elastic\\Agent\\data\\elastic-agent-dc443b\\components\\metricbeat.exe, possible process injection	14	92920
Defense Evasion, Privilege Escalation	Windows Remote Dektop utility process was accessed by C:\\Program Files\\Elastic\\Agent\\data\\elastic-agent-dc443b\\components\\metricbeat.exe, possible process injection	14	92920
Defense Evasion, Privilege Escalation	Explorer process was accessed by C:\\Program Files\\Elastic\\Agent\\data\\elastic-agent-dc443b\\components\\metricbeat.exe, possible process injection	12	92910

On peut ensuite cliquer dessus pour avoir le nom du fichier xml qui les gèrent :

Explorer process was accessed by [win.eventdata.sourceImage](#) , possible process injection [View alerts of this Rule](#)

Information

ID	Level	File	Path
92910	12	0945-sysmon_id_10.xml	ruleset/rules

Groups
sysmon, sysmon_eid10_detections, windows

Details

If_group	Win.Eventdata.TargetImage	Options
sysmon_event_10	pattern: (?i)explorer\.exe, type: pcre2	no_full_log

Compliance

Related rules

ID ↑	Description	Groups	Compliance	Level	File
------	-------------	--------	------------	-------	------

On peut alors se connecter sur notre serveur Wazuh via SSH puis aller chercher le fichier responsable :

```
find / -name 0945-sysmon_id_10.xml
```

```
root@Wazuh-Serv:~# find / -name 0945-sysmon_id_10.xml
/var/ossec/ruleset/rules/0945-sysmon_id_10.xml
```

On peut alors éditer le fichier :

```
nano /var/ossec/ruleset/rules/0945-sysmon_id_10.xml
```

Pour rajouter une exception, on va utiliser l'option negate en précisant quel exe il doit laisser passer :

```
<field name="win.eventdata.sourceImage" type="pcre2" negate="yes">(?  
i)metricbeat\.exe</field>
```

Nos règles passent donc de :

```
<rule id="92920" level="14">  
  <if_group>sysmon_event_10</if_group>  
  <field name="win.eventdata.targetImage" type="pcre2">(?!mstsc\.exe</field>
```

```

<options>no_full_log</options>
<description>Windows Remote Dektop utility process was accessed by
$(win.eventdata.sourceImage), possible process injection</description>
<mitre>
<id>T1055</id>
</mitre>
</rule>

```

à :

```

<rule id="92920" level="14">
  <if_group>sysmon_event_10</if_group>
  <field name="win.eventdata.targetImage" type="pcre2">(?!mstsc\.exe</field>
  <field name="win.eventdata.sourceImage" type="pcre2" negate="yes">(?!
i)metricbeat\.exe</field>
  <options>no_full_log</options>
  <description>Windows Remote Dektop utility process was accessed by
$(win.eventdata.sourceImage), possible process injection</description>
  <mitre>
  <id>T1055</id>
  </mitre>
</rule>

```

```

<rule id="92920" level="14">
  <if_group>sysmon_event_10</if_group>
  <field name="win.eventdata.targetImage" type="pcre2">(?!mstsc\.exe</field>
  <field name="win.eventdata.sourceImage" type="pcre2" negate="yes">(?!metricbeat\.exe</field>
  <options>no_full_log</options>
  <description>Windows Remote Dektop utility process was accessed by $(win.eventdata.sourceImage), possible process injection</description>
  <mitre>
    <id>T1055</id>
  </mitre>
</rule>
</group>

```

On peut alors terminer par redémarrer le serveur Wazuh :

```
systemctl restart wazuh-manager
```

Vu la quantité de faux-positifs générés et au vu de notre situation (pas en contexte de production), on peut alors supprimer tout les events pour repartir de zéro avec l'API intégré de Wazuh.

On peut lister nos index avec Curl :

```
curl -k -u admin:<mdp> https://10.202.0.98:9200/_cat/indices/wazuh-alerts*
```

```

root@Wazuh-Serv:~# curl -k -u admin:x66RYLr...U4x.RQ https://10.202.0.98:9200/_cat/indices/wazuh-alerts*
green open wazuh-alerts-4.x-2023.11.28 lKrw9BzQSWs0mu1xYQ0XFw 3 0 131197 0 135.7mb 135.7mb
green open wazuh-alerts-4.x-2023.11.29 qpIF-jDlRLWcZGxRHY4DCA 3 0 65482 0 65.7mb 65.7mb
green open wazuh-alerts-4.x-2023.11.24 YtGzmyKBTWysUXrU0gqrw 3 0 13022 0 17.3mb 17.3mb
green open wazuh-alerts-4.x-2023.11.25 333P-nQDSNa-B3E7Cg1foQ 3 0 40045 0 42.3mb 42.3mb
green open wazuh-alerts-4.x-2023.11.26 essxTTGMS5eB0tY2TEptDA 3 0 39893 0 44mb 44mb
green open wazuh-alerts-4.x-2023.11.27 sfgpDnHzTmiWNSBaJaNrew 3 0 89738 0 102.1mb 102.1mb
root@Wazuh-Serv:~#

```

On peut ensuite les supprimer en utilisant la méthode DELETE :

```
curl -k -X DELETE -u admin:<mdp> https://10.202.0.98:9200/wazuh-alerts-4.x-*
```

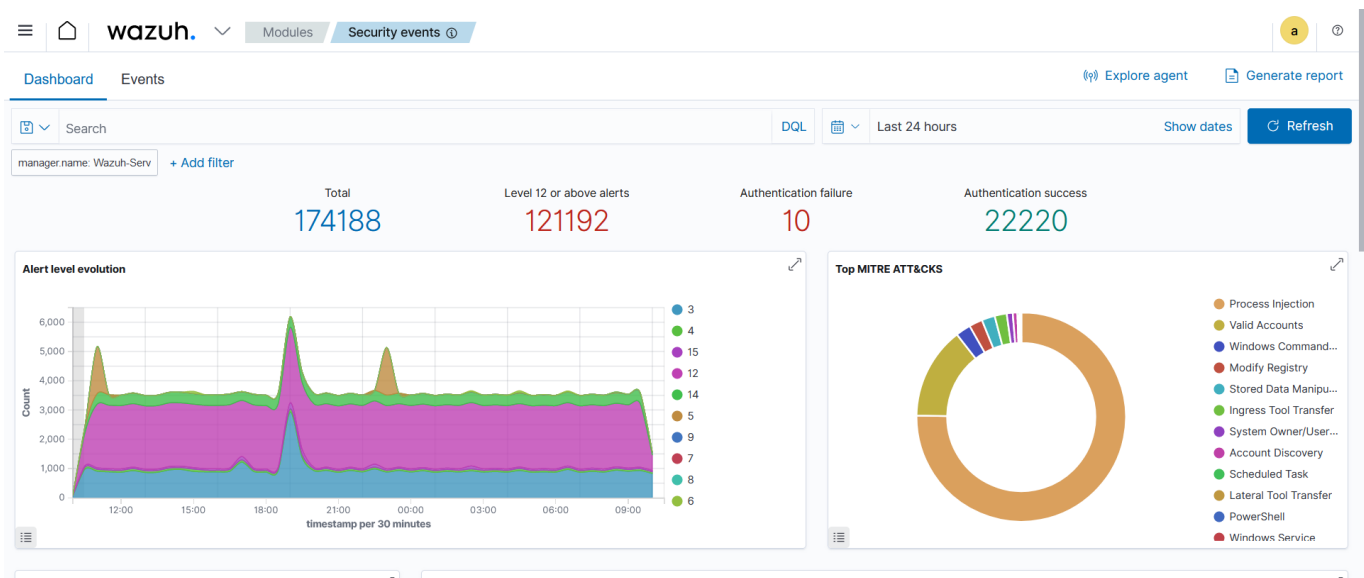
Ou en précisant qu'un index :

```
curl -k -X DELETE -u admin:<mdp> https://10.202.0.98:9200/wazuh-alerts-4.x-2023.11.28
```

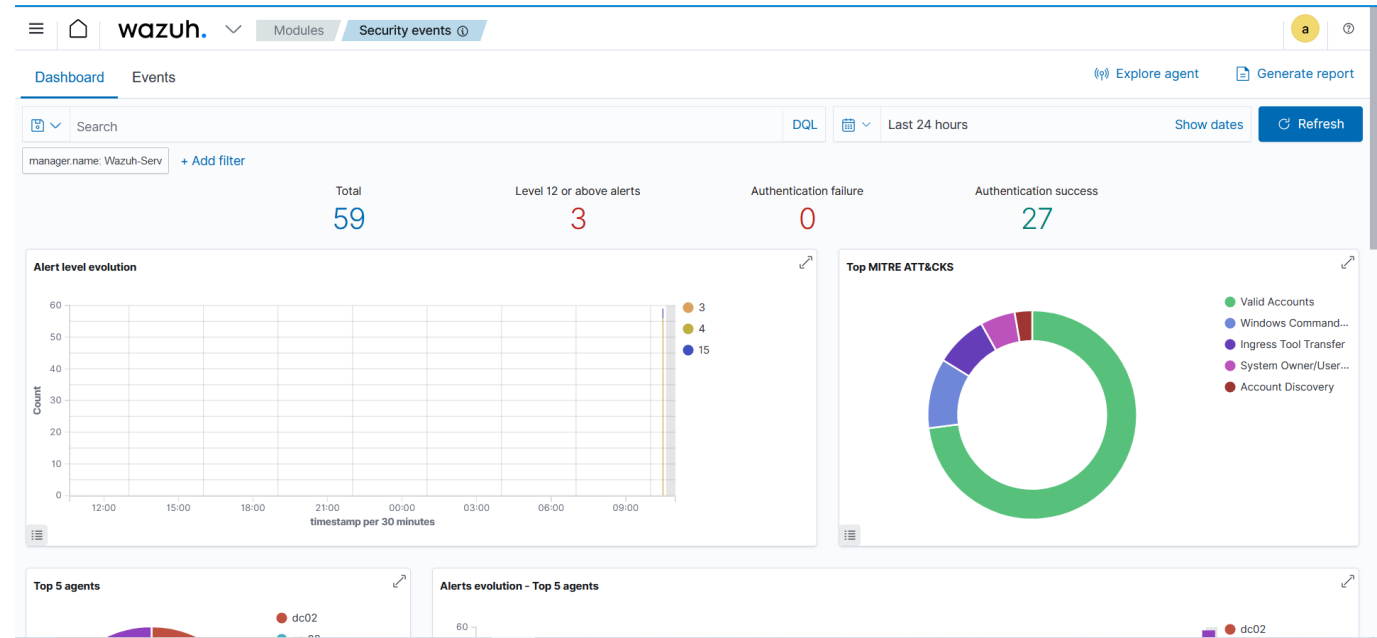
Si la commande est bonne, alors la réponse doit être :

```
{"acknowledged":true}
```

De retour sur le Dashboard Wazuh, on remarque que la méthode a fonctionné. On passe de :



à :



C'est mieux ! On a maintenant uniquement les alertes légitimes :

Nov 29, 2023 @ 10:38:24.740	004	dc02	T1105	Command and Control	Executable file dropped in folder commonly used by malware	15	92213
Table	JSON	Rule					
@timestamp		2023-11-29T09:38:24.740Z					
_id		RqFxGowBKT78MArisxwe					
agent.id		004					
agent.ip		10.202.0.118					
agent.name		dc02					
data.win.eventdata.creationUtcTime		2023-11-29 09:38:23.381					
data.win.eventdata.image		C:\Windows\System32\WindowsPowerShell\v1.0\powershell.exe					
data.win.eventdata.processGuid		{e295f343-068f-6567-f2f4-000000000c00}					
data.win.eventdata.processId		3324					
data.win.eventdata.ruleName		technique_id=T1059.001,technique_name=PowerShell					
data.win.eventdata.targetFilename		C:\Users\robb.stark\AppData\Local\Temp__PSScriptPolicyTest_geokel1r.1nm.ps1					
data.win.eventdata.user		NORTH\robb.stark					
data.win.eventdata.utcTime		2023-11-29 09:38:23.381					

On peut également se servir de cette méthode pour modifier / régler des règles déjà existantes pour qu'elles correspondent mieux à nos besoins, comme la détection d'attaque Bruteforce :

```
/var/ossec/ruleset/rules/0580-win-security_rules.xml
```

```

<rule id="60204" level="12" frequency="$MS_FREQ" timeframe="10">
  <if_matched_group>authentication_failed</if_matched_group>
  <same_field>win.eventdata.ipAddress</same_field>
  <options>no_full_log</options>
  <description>Plusieurs tentatives de connexions infructueuses en moins de 10 secondes. Bruteforce très probable
</description>
  <mitre>
    <id>T1110</id>
  </mitre>
  <group>authentication_failures,gdpr_IV_32.2,gdpr_IV_35.7.d,hipaa_164.312.b,nist_800_53_AC.7,nist_800_53_AU.14,nist_800_53_SI.4,pci_dss_10.2.4,pci_dss_10.2.5,pci_dss_11.4,tsc_CC6.1,tsc_CC6.8,tsc_CC7.2,tsc_CC7.3,</group>
</rule>

<rule id="60205" level="12" frequency="$MS_FREQ" timeframe="100">
  <if_matched_sid>60104</if_matched_sid>
  <same_field>win.eventdata.ipAddress</same_field>
  <options>no_full_log</options>
  <description>Multiple Windows audit failure events.</description>
  <group>gdpr_IV_35.7.d,hipaa_164.312.b,nist_800_53_AU.6,pci_dss_10.6.1,tsc_CC7.2,tsc_CC7.3,</group>
</rule>

<rule id="60206" level="12" frequency="$MS_FREQ" timeframe="240">
  <if_matched_sid>60102</if_matched_sid>
  <options>no_full_log</options>
  <description>Multiple Windows error security events.</description>
</rule>

<rule id="60207" level="12" frequency="$MS_FREQ" timeframe="120">
  <if_matched_sid>60101</if_matched_sid>
  <options>no_full_log</options>
  <description>Multiple Windows warning security events.</description>
</rule>
</group>

<!-- MS IPsec rules -->
<group name="windows,windows_security,ipsec,">
  <rule id="60208" level="8">

```

Ici, on a par exemple modifier l'état des alertes pour les passer de 10 à 12 (grave), la description de l'alerte qui s'affiche sur notre dashboard puis l'intervalle de temps en secondes sur laquelle il se concentre (timeframe) ainsi que la fréquence (par exemple 5 connexions ratées en moins de 10 secondes) :

```

→
<var name="MS_FREQ">5</var>
<group name="windows,windows_security,ipsec,">
  <rule id="60100" level="0">

```

On oublie évidemment pas de redémarrer notre service :

```
systemctl restart wazuh-manager.service
```

On peut aller encore plus loin dans notre démarche en utilisant la construction des données Wazuh pour récupérer directement l'IP de l'attaquant (si donné il ne l'a change pas à chaque tentatives). Sur notre Dashboard, on peut voir que les données sont construites sous format JSON comme ceci :

```

\ : \ north.sevenkingdoms.local \ , \ status \ : \ 0xc000006d \ , \ failureReason
\ , \ "authenticationPackageName" \ : \ "NTLM" \ , \ "keyLength" \ : \ "0" \ , \ "processId"
  "manager" : {
    "name" : "Wazuh-Serv"
  },
  "data" : {
    "win" : {
      "eventdata" : {
        "subjectLogonId" : "0x0",
        "ipAddress" : "10.202.0.172",
        "authenticationPackageName" : "NTLM",
        "subStatus" : "0xc000006a",
        "logonProcessName" : "NtLmSsp",
        "targetUserName" : "brandon.stark",
        "keyLength" : "0",
        "subjectUserSid" : "S-1-0-0",
        "processId" : "0x0",
        "ipPort" : "41184",
        "failureReason" : "%2313",
        "targetDomainName" : "north.sevenkingdoms.local",
        "targetUserSid" : "S-1-0-0",
        "logonType" : "3",
        "status" : "0xc000006d"
      },
      "system" : {
        "eventID" : "4625",
        "keywords" : "0x8010000000000000",
        "providerGuid" : "{54849625-5478-4994-a5ba-3e3b0328c30d}",

```

L'ip 10.202.0.172 est celle de l'attaquant. Il faudra alors afficher dans la description de l'attaque l'objet ipAddress de data => win => eventdata.

```
$(win.eventdata.ipAddress)
```

L'exemple d'une règle complète avec la variable :

```

<rule id="60204" level="12" frequency="$MS_FREQ" timeframe="10">
  <if_matched_group>authentication_failed</if_matched_group>
  <same_field>win.eventdata.ipAddress</same_field>
  <options>no_full_log</options>
  <description>Plusieurs tentatives de connexions infructueuses en moins de 10
secondes. Bruteforce très propable ! Attaquant : $(win.eventdata.ipAddress)
</description>
  <mitre>
    <id>T1110</id>
  </mitre>

```



```
<group>authentication_failures,gdpr_IV_32.2,gdpr_IV_35.7.d,hipaa_164.312.b,nist_800_53_AC.7,nist_800_53_AU.14,nist_800_53_SI.4,pci_dss_10.2.4,pci_dss_10.2.5,pci_dss_11.4,tsc_CC6.1,tsc_CC6.8,tsc_CC7.2,tsc_CC7.3,</group>
</rule>
```

Ce qui donne sur le Dashboard :

>	Dec 8, 2023 @ 09:08:28.975	004	dc02	T1110	Credential Access	Plusieurs tentatives de connexions infructueuses en moins de 10 secondes. Bruteforce très probable ! Attaquant : 10.202.0.172	12	60204
---	----------------------------	-----	------	-------	-------------------	---	----	-------

Activation de l'Active Response avec Wazuh :

Malgré tout, dans son fonctionnement initial, Wazuh agit uniquement comme un IDS (Intrusion Detection System) et ne bloque donc pas activement les menaces. Il faut alors rajouter de nouvelles règles pour les bloquer.

On commence par regarder si l'entrée firewall-drop (blocage IP de l'attaquant avec la création de règles IPTables) dans le fichier suivant :

```
/var/ossec/etc/ossec.conf
```

```
<command>
  <name>firewall-drop</name>
  <executable>firewall-drop</executable>
  <timeout_allowed>yes</timeout_allowed>
</command>

<command>
  <name>host-deny</name>
```

Maintenant, toujours dans le même fichier, on va passer à la rédaction de règles. Ici, le concept est simple :

```
<ossec_config>
  <active-response>
    <command>firewall-drop</command>
    <location>local</location>
    <rules_id>5551</rules_id>
    <timeout>180</timeout>
```

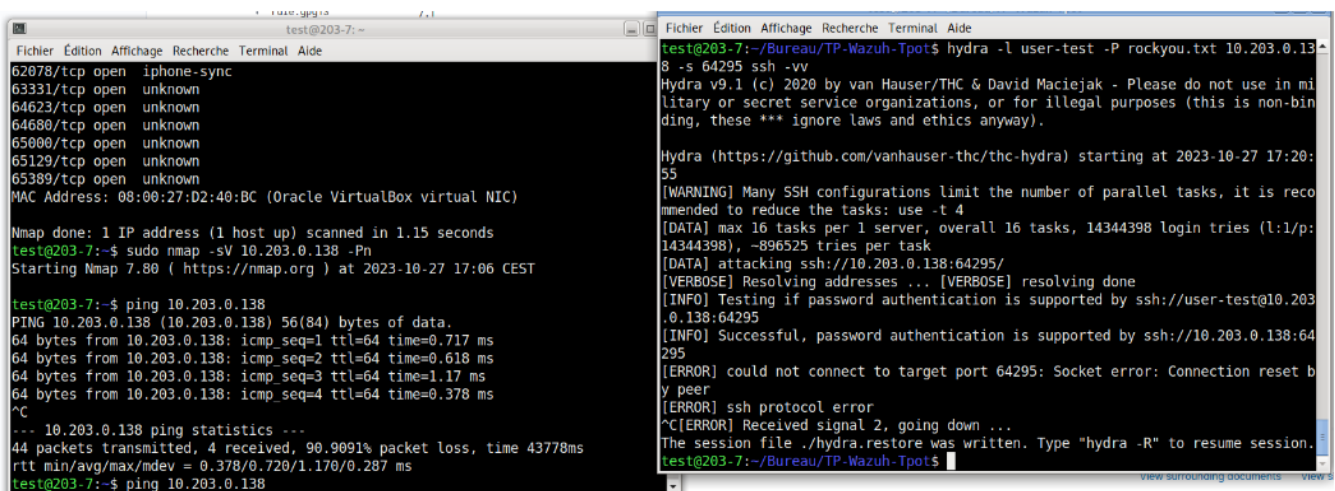
```
</active-response>
</ossec_config>
```

On commence par définir la commande que le serveur va lancer (ici, firewall-drop qui ban IP l'attaquant), puis la règle qui va faire lancer la réaction (5551 = plusieurs tentatives de connexions infructueuses) puis le timeout (durée du bannissement de l'attaquant, en secondes).

```
sudo systemctl restart wazuh-manager
```

On oublie évidemment pas de relancer le serveur à chaque modifications...

On peut tester la réaction du serveur en lançant une attaque bruteforce sur le SSH de la machine avec hydra pour voir si la réponse fonctionne en réalisant des pings en continu :



The image shows two terminal windows. The left window, titled 'test@203-7: ~', displays the output of an Nmap scan and a series of pings to 10.203.0.138. The Nmap scan shows a single IP address scanned in 1.15 seconds. The ping tests show successful responses with varying times. The right window, titled 'Fichier Édition Affichage Recherche Terminal Aide', shows a Hydra brute-force attack on the SSH service of 10.203.0.138. The attack is configured with the username 'user-test' and a password file 'rockyou.txt'. The output shows the Hydra version (v9.1), the target IP and port (10.203.0.138:64295), and the results of the attack, including a warning about SSH configurations, a successful password authentication, and a subsequent connection reset error.

```
test@203-7: ~
Fichier Édition Affichage Recherche Terminal Aide
62078/tcp open  iphone-sync
63331/tcp open  unknown
64623/tcp open  unknown
64680/tcp open  unknown
65000/tcp open  unknown
65129/tcp open  unknown
65389/tcp open  unknown
MAC Address: 08:00:27:D2:40:BC (Oracle VirtualBox virtual NIC)
Nmap done: 1 IP address (1 host up) scanned in 1.15 seconds
test@203-7:~$ sudo nmap -sV 10.203.0.138 -Pn
Starting Nmap 7.80 ( https://nmap.org ) at 2023-10-27 17:06 CEST

test@203-7:~$ ping 10.203.0.138
PING 10.203.0.138 (10.203.0.138) 56(84) bytes of data:
64 bytes from 10.203.0.138: icmp_seq=1 ttl=64 time=0.717 ms
64 bytes from 10.203.0.138: icmp_seq=2 ttl=64 time=0.618 ms
64 bytes from 10.203.0.138: icmp_seq=3 ttl=64 time=1.17 ms
64 bytes from 10.203.0.138: icmp_seq=4 ttl=64 time=0.378 ms
^C
--- 10.203.0.138 ping statistics ---
44 packets transmitted, 4 received, 90.9091% packet loss, time 43778ms
rtt min/avg/max/mdev = 0.378/0.720/1.170/0.287 ms
test@203-7:~$ ping 10.203.0.138

test@203-7:~/Bureau/TP-Wazuh-Tpot$ hydra -l user-test -P rockyou.txt 10.203.0.138 -s 64295 ssh -vv
Hydra v9.1 (c) 2020 by van Hauser/THC & David Maciejak - Please do not use in military or secret service organizations, or for illegal purposes (this is non-binding, these *** ignore laws and ethics anyway).

Hydra (https://github.com/vanhauser-thc/thc-hydra) starting at 2023-10-27 17:20:55
[WARNING] Many SSH configurations limit the number of parallel tasks, it is recommended to reduce the tasks: use -t 4
[DATA] max 16 tasks per 1 server, overall 16 tasks, 14344398 login tries (l:1/p:14344398), ~896525 tries per task
[DATA] attacking ssh://10.203.0.138:64295/
[VERBOSE] Resolving addresses ... [VERBOSE] resolving done
[INFO] Testing if password authentication is supported by ssh://user-test@10.203.0.138:64295
[INFO] Successful, password authentication is supported by ssh://10.203.0.138:64295
[ERROR] could not connect to target port 64295: Socket error: Connection reset by peer
[ERROR] ssh protocol error
^C[ERROR] Received signal 2, going down ...
The session file ./hydra.restore was written. Type "hydra -R" to resume session.
test@203-7:~/Bureau/TP-Wazuh-Tpot$
```

Notre règle fonctionne bien ! Le ping ne passe plus lorsqu'on lance l'attaque avec hydra.