Başlarken

Bu kılavuz, Wazuh sunucu API'sini kullanmak için gereken temel bilgileri sağlar.

Wazuh Sunucu API'sini Başlatma ve Durdurma

Wazuh yöneticisini yüklediğinizde, Wazuh sunucu API'si de sürecin bir parçası olarak varsayılan olarak yüklenir. Wazuh yönetici hizmetiyle `systemctl` veya komutlarını yürüterek Wazuh sunucu API'sini yönetebilir veya izleyebilirsiniz: `service`

Systemd

```
systemctl start/status/stop/restart wazuh-manager
```

SysV Başlatma

```
service wazuh-manager start/status/stop/restart
```

Wazuh Dashboard Aracılığıyla Wazuh Sunucu API'sini Kullanma

Wazuh panosu aracılığıyla Wazuh sunucu API'siyle etkileşim kurabilirsiniz. Bunu yapmak için, yönetici ayrıcalıklarına sahip bir kullanıcıyla Wazuh panosuna giriş yapmanız gerekir. Örneğin, varsayılan `admin` kullanıcı yönetici ayrıcalıklarına sahiptir. Panodaki Wazuh sunucu API konsoluna erişmek için menü simgesine tıklayın ve **Araçlar** > **API Konsolu'na** gidin .

Pano üzerinden Wazuh sunucu API konsoluna erişin

Image not found or type unknown

API Konsolu'nda yöntemi , istek uç noktasını ve herhangi bir sorgu parametresini girin, ardından isteği yürütmek için oynat düğmesine tıklayın. Temel kavramlar hakkında daha fazla bilgi edinmek için Wazuh sunucusu API isteğini ve yanıtını anlama bölümüne bakın.

API isteğini yürüt

Image not found or type unknown

Komut Satırı Aracılığıyla Wazuh Sunucu API'sine Giriş Yapma

Güvenli erişimi sağlamak için tüm Wazuh sunucu API uç noktaları kimlik doğrulaması gerektirir. Kullanıcılar her istekte bir JSON Web Token (JWT) eklemelidir. JWT, taraflar arasında bilgileri bir JSON nesnesi olarak güvenli bir şekilde iletmek için kompakt ve kendi kendine yeten bir yöntem tanımlayan açık bir standarttır (RFC 7519). POST /security/user/authenticate kullanarak Wazuh sunucu API'sine giriş yapmak ve API uç noktalarına erişmek için gerekli bir belirteç edinmek için aşağıdaki adımları izleyin:

1. Wazuh sunucu API'sine bir kullanıcı kimlik doğrulama POST isteği göndermek ve döndürülen JWT'yi değişkende depolamak için aşağıdaki komutu çalıştırın `TOKEN`. `<WAZUH_API_USER>`ve'yi `<WAZUH_API_PASSWORD>`kimlik bilgilerinizle değiştirin.

   ```
   TOKEN=$(curl -u <WAZUH_API_USER>:<WAZUH_API_PASSWORD> -k -X POST "https://localhost:55000/security/user/authenticate?raw=true")
   ```

   > Not:
   > - `SSL`API'de (HTTPS) etkinleştirilmişse ve varsayılan kendi kendine imzalanmış sertifikaları kullanıyorsa, sunucu bağlantı doğrulamasını önlemek için parametreyi eklemeniz gerekir . cURL komutları aracılığıyla kimlik doğrulaması yaparken sorgu parametresini `-k`kullanmanızı öneririz `raw=true`, çünkü belirteci düz metin olarak döndürerek işlemeyi basitleştirir, özellikle uzun JWT'ler için yararlıdır.
   >
   > Varsayılan Wazuh sunucusu API kimlik bilgisi 'dir `wazuh:wazuh`. Ancak Wazuh dağıtımı kurulum betiği kullanılarak gerçekleştirildiyse, Wazuh API kullanıcısı 'dir ve ' komutunu çalıştırarak `wazuh`parolayı çıkarabilirsiniz .`wazuh-install-files.tar -axf wazuh-install-files.tar wazuh-install-files/wazuh-passwords.txt -O | grep -P "\'wazuh\'" -A 1`
   >
   > Eğer `wazuh` şifrenizi geri alamazsanız, kullanıcı şifresini sıfırlayabilirsiniz .

2. Jetonun oluşturulduğunu doğrulayın:

   ```
   echo $TOKEN
   ```

   Çıktı aşağıdakine benzer uzun bir dize olmalıdır:

   Output

   ```
   eyJhbGciOiJFUzUxMiIsInR5cCI6IkpXVCJ9.eyJpc3MiOiJ3YXp1aCIsImF1ZCI6IldhenVoIEFQSSBSRVNUIiwibmJmIjoxM
   ```

Kimlik doğrulama başarısız olursa, çıktı bir hata mesajı görüntüler veya boş kalır. Bu gibi durumlarda, kullanıcı kimlik bilgilerinizi iki kez kontrol edin ve Wazuh sunucu API'sine ağ bağlantınız olduğundan emin olun.

3. Her şeyin beklendiği gibi çalıştığını doğrulamak için bir API isteği gönderin:

```
curl -k -X GET "https://localhost:55000/" -H "Authorization: Bearer $TOKEN"
```

Output

```
{
   "data": {
      "title": "Wazuh API REST",
      "api_version": "4.7.4",
      "revision": 40717,
      "license_name": "GPL 2.0",
      "license_url": "https://github.com/wazuh/wazuh/blob/master/LICENSE",
      "hostname": "wazuh-master",
      "timestamp": "2024-05-14T21:34:15Z"},
  "error": 0
}
```

Giriş yaptıktan sonra, aşağıdaki yapıyı kullanarak herhangi bir API uç noktasına erişebilirsiniz. `<METHOD>`İstediğiniz yöntemle ve `<ENDPOINT>`erişmek istediğiniz uç noktaya karşılık gelen dizeyle değiştirin. Bir ortam değişkeni kullanmıyorsanız, `$TOKEN`elde edilen JWT ile değiştirin.

```
curl -k -X <METHOD> "https://localhost:55000/<ENDPOINT>" -H  "Authorization: Bearer $TOKEN"
```

Scriptler Aracılığıyla Wazuh Sunucu API'sine Giriş Yapma

Bu bölüm, Wazuh sunucusuyla etkileşimleri otomatikleştirmek için temel bir adım olan komut dosyalarını kullanarak Wazuh sunucusu API'sine giriş yapma sürecini ayrıntılı olarak açıklar. Sağlanan örnekler, hem varsayılan (`false`) hem de düz metin (`true`) `raw`parametrelerini göstererek gerçek dünya uygulamalarını sergilemeyi amaçlamaktadır. `raw`Parametre, olarak ayarlandığında `true` , yanıtın düz metin veya asgari düzeyde işlenmiş bir biçimde olması gerektiği anlamına gelir. Tersine, `raw`parametre olduğunda `false`, yanıt ayrıştırma ve entegrasyonu kolaylaştırmak için daha yapılandırılmış bir JSON biçimindedir. Bu komut dosyaları, otomasyon yoluyla operasyonel verimliliğini artırmak isteyen veya özel entegrasyonlar için Wazuh sunucusu API'sine programlı olarak nasıl erişileceğini anlamak isteyen kullanıcılar için tasarlanmıştır.

Python Scriptiyle Oturum Açma

Python betiği kullanarak Wazuh sunucu API'sine kimlik doğrulaması yapabilirsiniz. Aşağıdaki betik, `wazuh_api_authenticator.py`bir JWT elde etmek için Wazuh sunucu API'siyle kimlik doğrulaması yapar. Daha sonra Wazuh aracılarının durumlarının bir özetini almak için istek başlığındaki belirteci kullanır.

```
#!/usr/bin/env python3

import json
import requests
import urllib3
from base64 import b64encode

# Disable insecure https warnings (for self-signed SSL certificates)
urllib3.disable_warnings(urllib3.exceptions.InsecureRequestWarning)

# Configuration
protocol = 'https'
host = 'localhost'
port = 55000
user = '<WAZUH_API_USER>'
password = '<WAZUH_API_PASSWORD>'
login_endpoint = 'security/user/authenticate'

login_url = f"{protocol}://{host}:{port}/{login_endpoint}"
basic_auth = f"{user}:{password}".encode()
login_headers = {'Content-Type': 'application/json',
                 'Authorization': f'Basic {b64encode(basic_auth).decode()}'}

print("\nLogin request ...\n")
response = requests.post(login_url, headers=login_headers, verify=False)
token = json.loads(response.content.decode())['data']['token']
print(token)

# New authorization header with the JWT we got
requests_headers = {'Content-Type': 'application/json',
                    'Authorization': f'Bearer {token}'}

print("\n- API calls with TOKEN environment variable ...\n")

print("Getting API information:")

response = requests.get(f"{protocol}://{host}:{port}/?pretty=true", headers=requests_headers, verify=False)
print(response.text)

print("\nGetting agents status summary:")

response = requests.get(f"{protocol}://{host}:{port}/agents/summary/status?pretty=true", headers=requests_he
print(response.text)
```

```
print("\nEnd of the script.\n")
```

`<WAZUH_API_USER>`ve `<WAZUH_API_PASSWORD>`ifadelerini doğru bilgilerle değiştirin .

Python `requests`modülünü kurun:

```
python3 -m pip install requests
```

> Not: Python modülü `urllib3`sürüm 2.0 ve üzeri yalnızca OpenSSL sürüm 1.1.1 veya üzerini destekler. Sisteminizde daha eski bir OpenSSL sürümü varsa, şunlardan birini yapmanız gerekir:
>
> - OpenSSL'i 1.1.1 veya daha üst bir sürüme yükseltin.
> - `urllib3`Mevcut OpenSSL sürümünüzle uyumlu bir sürüme geçin .

Uyumluluk sorunlarını önlemek için lütfen yazılım bağımlılıklarınızın düzgün bir şekilde hizalandığından emin olun.

Python betiğini çalıştırın `wazuh_api_authenticator.py`:

```
python3 wazuh_api_authenticator.py
```

Output

```
Login request ...
eyJ0eXAiOiJKV1QiLCJhbGciOiJIUzI1NiJ9.eyJpc3MiOiJ3YXp1aCIsImF1ZCI6IldhenVoIEFQSSBSRVNUIiwibmJmIjoxNjAyMj
- API calls with TOKEN environment variable ...
Getting API information:
{
  "data": {
    "title": "Wazuh API REST",
    "api_version": "4.7.4",
    "revision": 40717,
    "license_name": "GPL 2.0",
    "license_url": "https://github.com/wazuh/wazuh/blob/master/LICENSE",
    "hostname": "wazuh-master",
    "timestamp": "2024-05-14T21:34:15Z"
  },
  "error": 0
}
Getting agents status summary:
{
  "data": {
     "connection": {
        "active": 1,
        "disconnected": 0,
        "never_connected": 0,
        "pending": 0,
```

```
      "total": 1
    },
    "configuration": {
      "synced": 1,
      "not_synced": 0,
      "total": 1
    }
  },
  "error": 0
}
End of the script.
```

Bash Script Oturum Açma

Ayrıca bir Bash betiği kullanarak Wazuh sunucu API'sine kimlik doğrulaması yapabilirsiniz. Aşağıdaki betik, `wazuh_api_authenticator.sh`bir JWT elde etmek için Wazuh sunucu API'siyle kimlik doğrulaması yapar. Daha sonra Wazuh aracıları tarafından kullanılan işletim sistemlerinin bir özetini almak için istek başlığındaki belirteci kullanır.

```
#!/bin/bash

echo -e "\n- Getting token...\n"

TOKEN=$(curl -u <WAZUH_API_USER>:<WAZUH_API_PASSWORD> -k -X POST "https://localhost:55000/security/us

echo -e "\n- API calls with TOKEN environment variable ...\n"

echo -e "Getting default information:\n"

curl -k -X GET "https://localhost:55000/?pretty=true" -H  "Authorization: Bearer $TOKEN"

echo -e "\n\nGetting /agents/summary/os:\n"

curl -k -X GET "https://localhost:55000/agents/summary/os?pretty=true" -H  "Authorization: Bearer $TOKEN"

echo -e "\n\nEnd of the script.\n"
```

`<WAZUH_API_USER>`ve `<WAZUH_API_PASSWORD>`öğesini doğru kimlik bilgileriyle değiştirin

Bash betiğini çalıştırın `wazuh_api_authenticator.sh`:

```
bash wazuh_api_authenticator.sh
```

Output

```
- Getting token...
Total    % Received % Xferd  Average Speed   Time    Time     Time  Current
                             Dload  Upload   Total   Spent    Left  Speed
100  3059  100  3059    0     0  17089      0 --:--:-- --:--:-- --:--:-- 17089
- API calls with TOKEN environment variable ...
```

```
Getting default information:
{
  "data": {
    "title": "Wazuh API REST",
    "api_version": "4.7.4",
    "revision": 40717,
    "license_name": "GPL 2.0",
    "license_url": "https://github.com/wazuh/wazuh/blob/master/LICENSE",
    "hostname": "wazuh-master",
    "timestamp": "2024-05-14T21:34:15Z"
  },
  "error": 0
}
Getting /agents/summary/os:
{
   "data": {
      "affected_items": [
         "windows"
      ],
      "total_affected_items": 1,
      "total_failed_items": 0,
      "failed_items": []
   },
   "message": "Showing the operative system of all specified agents",
   "error": 0
}
End of the script.
```

Wazuh Sunucusu API İsteğini ve Yanıtını Anlama

Standart bir Wazuh sunucusu API isteği üç temel bileşenden oluşur: istek yöntemi (GET, POST, PUT veya DELETE), erişilen uç noktayı belirten API URL'si ve yetkilendirme başlığı. Bu başlık, isteği doğrulamak ve yetkilendirmek için bir JWT içermelidir. Aşağıda bir cURL isteği örneği verilmiştir:

```
curl -k -X GET "https://localhost:55000/agents/summary/os?pretty=true" -H  "Authorization: Bearer $TOKEN"
```

Her istek için cURL komutu aşağıdaki alanları içerir:

Alan	Tanım
`-X GET/POST/PUT/DELETE`	HTTP sunucusuyla iletişim kurarken kullanılacak bir istek yöntemi belirtin.
`http://<WAZUH_MANAGER_IP>:55000/<ENDPOINT>` `https://<WAZUH_MANAGER_IP>:55000/<ENDPOINT>`	Kullanılacak API URL'si. API'de SSL'nin aktif olup olmadığına bağlı olarak `http`belirtin .`https`

Alan	Tanım
`-H "Authorization: Bearer <YOUR_JWT_TOKEN>"`	JWT'yi belirtmek için isteğe ek bir başlık ekleyin.
`-k`	SSL sertifika hatalarını bastırın (yalnızca varsayılan kendi kendine imzalı sertifikaları kullanıyorsanız).

Tüm yanıtlar JSON formatındadır ve çoğu şu yapıyı takip eder:

Alan	İsteğe bağlı alt alanlar	Tanım
veri	etkilenen_öğeler	İstekte başarıyla etkilenen öğelerin her birini listeleyin.
	toplam_etkilenen_öğeler	Başarıyla etkilenen öğelerin toplam sayısı.
	başarısız_öğeler	İstekteki başarısız olan her öğeyi içeren liste.
	toplam_başarısız_öğeler	Başarısız olan öğelerin toplam sayısı.
mesaj		Sonuç açıklaması.
hata		HTTP yanıtları için `200`, yanıtın tamamlanmış (`0`), başarısız (`1`) veya kısmi (`2`) olup olmadığını belirler. HTTP `4xx`veya `5xx`yanıtları için, başarısızlıkla ilgili hata kodunu belirler.

- Varsayılan olarak, veri koleksiyonları içeren yanıtlar en fazla 500 öğe döndürür. Büyük koleksiyonlar arasında yineleme yapmak için `offset`ve parametrelerini kullanabilirsiniz . Parametre 100.000 öğeye kadar izin verse de, zaman aşımı ve aşırı büyük yanıtlar gibi beklenmeyen davranışları önlemek için varsayılan 500 öğe sınırını aşmamanızı öneririz. Dikkatli kullanın.`limit``limit`
- Tüm yanıtlar bir HTTP durum kodu içerir: 2xx (başarılı), 4xx (istemci hatası), 5xx (sunucu hatası), vb.
- Tüm istekler (ve hariç) JSON yanıtını daha okunabilir bir biçime dönüştürmek için parametreyi kabul eder .`POST /security/user/authenticate``POST /security/user/authenticate/run_as``pretty`
- Wazuh sunucu API'si, seçilen günlük biçimine bağlı olarak günlükleri `api.log`veya `api.json` dosyalarında depolar. Bu günlük dosyaları Wazuh sunucusunda bulunur . Wazuh API yapılandırma dosyasında`/var/ossec/logs/` ayrıntı düzeyini değiştirebilirsiniz .
- Wazuh API günlükleri varsayılan olarak zamana göre döndürülür. Döndürme yalnızca günlüğe yeni bir giriş eklendikten sonra gerçekleşir. Örneğin, zaman tabanlı döndürme, her gece yarısı olmasa da farklı bir günde yeni bir giriş eklendiğinde tetiklenir. Döndürülmüş günlükler . `/var/ossec/logs/api/<year>/<month>/`kullanılarak depolanır ve sıkıştırılır `gzip`.
- `request_timeout`Sunucu API yapılandırma dosyasının alanında tanımlanan zaman süresinden sonra yanıt alınmazsa tüm Wazuh sunucu API istekleri iptal edilir `/var/ossec/api/configuration/api.yaml`. Bu zaman aşımını devre dışı bırakmak için parametreyi kullanabilirsiniz ; bu, özellikle PUT /agents/upgrade`wait_for_complete` gibi beklenen süreyi aşabilecek çağrılar için yararlıdır .

> Not: Maksimum API yanıt süresini ayarlamak için Wazuh sunucusundaki dosyadaki `request_timeout` değeri güncelleyin. `/var/ossec/api/configuration/api.yaml`

Hata içermeyen örnek yanıt (HTTP durum kodu 200):

Output


```
{
 "data": {
  "affected_items": [
   "master-node",
   "worker1"
  ],
  "total_affected_items": 2,
  "failed_items": [],
  "total_failed_items": 0
 },
 "message": "Restart request sent to all specified nodes",
 "error": 0
}
```


Hatalı örnek yanıt (HTTP durum kodu 200):

Output


```
{
 "data": {
  "affected_items": [],
  "total_affected_items": 0,
  "total_failed_items": 4,
  "failed_items": [
   {
    "error": {
     "code": 1707,
     "message": "Cannot send request, agent is not active",
     "remediation": "Please, check non-active agents connection and try again. Visit
     https://documentation.wazuh.com/current/user-manual/registering/index.html and
     https://documentation.wazuh.com/current/user-manual/agents/agent-connection.html
     to obtain more information on registering and connecting agents"
    },
    "id": [
     "001",
     "002",
     "009",
     "010"
    ]
   },
  ]
 },
 "message": "Restart command was not sent to any agent",
 "error": 1
}
```

Kısmi yanıt örneği (HTTP durum kodu 200):

Output


```
{
 "data": {
  "affected_items": [
   {
    "ip": "10.0.0.9",
    "id": "001",
    "name": "Carlos",
    "dateAdd": "2020-10-07T08:14:32Z",
    "node_name": "unknown",
    "registerIP": "10.0.0.9",
    "status": "never_connected"
   }
  ],
  "total_affected_items": 1,
  "total_failed_items": 1,
  "failed_items": [
   {
    "error": {
     "code": 1701,
     "message": "Agent does not exist",
     "remediation": "Please, use `GET /agents?select=id,name` to find all available agents"
    },
    "id": [
     "005"
    ]
   }
  ]
 },
 "message": "Some agents information was not returned",
 "error": 2
}
```


Yetkisiz bir isteği bildirmek için örnek yanıt (HTTP durum kodu 401):

Output


```
{
 "title": "Unauthorized",
 "detail": "The server could not verify that you are authorized to access the URL requested. You either supplied th
}
```


İzin reddedildi hatasını (HTTP durum kodu 403) bildirmek için örnek yanıt:

Output


```
{
 "title": "Permission Denied",
 "detail": "Permission denied: Resource type: *:*",
 "remediation": "Please, make sure you have permissions to execute the current request. For more information on
 "error": 4000,
```

```
  "dapi_errors": {
    "unknown-node": {
      "error": "Permission denied: Resource type: *:*"
    }
  }
}
```

Wazuh Sunucu API Kullanımına İlişkin Pratik Örnekler

Bu bölümde, cURL, Python betikleri ve PowerShell betikleri kullanarak Wazuh sunucu API'sine çeşitli istek türlerinin nasıl gönderileceğini gösteriyoruz. Bu örnekler, öngörebileceğiniz daha gelişmiş kullanım durumları için temel bilgi görevi görür.

CURL

cURL, HTTP/HTTPS istekleri ve komutları göndermek için bir komut satırı aracıdır. Birçok Linux ve macOS uç noktasına önceden yüklenmiş olarak gelir ve kullanıcıların Wazuh sunucu API'siyle doğrudan komut satırından etkileşim kurmasını sağlar. Herhangi bir uç noktayı yürütmeden önce bir JWT edinmeniz gerektiğini unutmayın. Aşağıdaki örneklerde, belirteci almak ve onu bir ortam değişkeni (`$TOKEN`) olarak kaydetmek için ham seçeneğini kullanıyoruz. JWT edinmeyle ilgili ayrıntılı talimatlar için lütfen başlarken bölümüne bakın.

GET

Aşağıdaki GET isteği, Wazuh sunucu API'si hakkında başlık, sürüm, revizyon, lisans, ana bilgisayar adı ve geçerli zaman damgası gibi temel bilgileri alır:

```
# curl -k -X GET "https://localhost:55000/" -H  "Authorization: Bearer $TOKEN"
```

Output

```
{
   "data": {
      "title": "Wazuh API",
      "api_version": "4.7.4",
      "revision": 40717,
      "license_name": "GPL 2.0",
      "license_url": "https://github.com/wazuh/wazuh/blob/master/LICENSE",
      "hostname": "wazuh-master",
      "timestamp": "2024-05-14T21:34:15Z"
   },
   "error": 0
}
```

POST

Wazuh sunucusu API'sine yapılan aşağıdaki POST isteği, istek gövdesinde kullanıcı adı `test_user`ve parola belirtilerek Wazuh sunucusunda yeni bir kullanıcı oluşturur .`Test_user1`

```
curl -k -X POST "https://localhost:55000/security/users" -H  "Authorization: Bearer $TOKEN" -H  "Content-Type: application/json" -d "{\"username\":\"test_user\",\"password\":\"Test_user1\"}"
```

Output


```
{
 "data": {
  "affected_items": [
   {
    "username": "test_user",
    "roles": []
   }
  ],
  "total_affected_items": 1,
  "total_failed_items": 0,
  "failed_items": []
 },
 "message": "User was successfully created",
 "error": 0
}
```


DELETE

Wazuh sunucusu API'sine gönderilen aşağıdaki DELETE isteği, Wazuh sunucusundaki tüm aracı gruplarını siler.

```
curl -k -X DELETE "https://localhost:55000/groups?pretty=true&groups_list=all" -H  "Authorization: Bearer $TOKEN"
```

Output


```
{
 "data": {
  "affected_items": [
   "group1",
   "group2",
   "group3"
  ],
  "total_affected_items": 3,
  "total_failed_items": 0,
  "failed_items": [],
  "affected_agents": [
   "001",
   "002",
```

```
      "003",
      "005",
      "006",
      "007",
      "008",
      "009",
      "010"
    ]
  },
  "message": "All selected groups were deleted",
  "error": 0
}
```


Python

Bağlantısı kesilen aracılar hakkında, son canlı tutma süreleri ve kimlikleri dahil olmak üzere bilgi almak için bir Python betiği kullanabilirsiniz. Bunu yapmak için, betik önce bir taşıyıcı belirteci almak için temel kimlik doğrulamasını kullanarak Wazuh sunucu API'siyle kimlik doğrulaması yapar, ardından gerekli bilgileri almak için bir GET isteği yapar.

Aşağıdaki Python betiğini şu şekilde kaydedin `get_agent_keep_alive.py`:

```
#!/usr/bin/env python3

import json
from base64 import b64encode

import requests  # To install requests, use: pip install requests
import urllib3

# Configuration
endpoint = '/agents?select=lastKeepAlive&select=id&status=disconnected'

protocol = 'https'
host = '<WAZUH_SERVER_API_IP>'
port = '<WAZUH_SERVER_API_PORT>'
user = '<WAZUH_API_USER>'
password = '<WAZUH_API_PASSWORD>'

# Disable insecure https warnings (for self-signed SSL certificates)
urllib3.disable_warnings(urllib3.exceptions.InsecureRequestWarning)

# Functions
def get_response(request_method, url, headers, verify=False, body=None):
    """Get API result"""
    if body is None:
        body = {}

    request_result = getattr(requests, request_method.lower())(url, headers=headers, verify=verify, data=body)
```

```
    if request_result.status_code == 200:
        return json.loads(request_result.content.decode())
    else:
        raise Exception(f"Error obtaining response: {request_result.json()}")

# Variables
base_url = f"{protocol}://{host}:{port}"
login_url = f"{base_url}/security/user/authenticate"
basic_auth = f"{user}:{password}".encode()
headers = {
          'Authorization': f'Basic {b64encode(basic_auth).decode()}',
          'Content-Type': 'application/json'
          }
headers['Authorization'] = f'Bearer {get_response("POST", login_url, headers)["data"]["token"]}'

# Request
response = get_response("GET", url=base_url + endpoint, headers=headers)

# WORK WITH THE RESPONSE AS YOU LIKE
print(json.dumps(response, indent=4, sort_keys=True))
```

Aşağıdaki değişkenleri değiştirin:

- `<WAZUH_SERVER_API_IP>`Wazuh sunucunuzun IP adresi ile.
- `<WAZUH_SERVER_API_PORT>`Wazuh sunucusu API port numarasıyla (varsayılan olarak port 5500).
- `<WAZUH_API_USER>`ve `<WAZUH_API_PASSWORD>`doğru belgelerle.

Python `requests`modülünü kurun:

```
python3 -m pip install requests
```

> Not: Python modülü `urllib3`sürüm 2.0 ve üzeri yalnızca OpenSSL sürüm 1.1.1 veya üzerini destekler. Sisteminizde daha eski bir OpenSSL sürümü varsa, şunlardan birini yapmanız gerekir:
> - OpenSSL'i 1.1.1 veya daha üst bir sürüme yükseltin.
> - `urllib3`Mevcut OpenSSL sürümünüzle uyumlu bir sürüme geçin.
>
> Uyumluluk sorunlarını önlemek için lütfen yazılım bağımlılıklarınızın düzgün bir şekilde hizalandığından emin olun.

Bağlantısı kesilen aracılar hakkında bilgi almak için Python betiğini çalıştırın:

```
python3 get_agent_keep_alive.py
```

Output

```
{
  "data": {
```

```
      "affected_items": [
         {
            "id": "009",
            "lastKeepAlive": "2020-05-23T12:39:50Z"
         },
         {
            "id": "010",
            "lastKeepAlive": "2020-05-23T12:39:50Z"
         }
      ],
      "failed_items": [],
      "total_affected_items": 2,
      "total_failed_items": 0
   },
   "message": "All selected agents information was returned",
   "error": 0
}
```


Güç Kabuğu

Bağlantısı kesilen aracıların son canlı tutma süreleri ve kimlikleri dahil olmak üzere ayrıntıları almak için bir PowerShell betiği de kullanabilirsiniz. Bunu yapmak için, betik önce bir taşıyıcı belirteci almak için temel kimlik doğrulamasını kullanarak Wazuh sunucu API'siyle kimlik doğrulaması yapar, ardından gerekli bilgileri almak için bir GET isteği yapar.

Aşağıdaki PowerShell betiğini şu şekilde kaydedin `get_agent_keep_alive.ps1`:

```
function Ignore-SelfSignedCerts {
   add-type @"
      using System.Net;
      using System.Security.Cryptography.X509Certificates;

      public class PolicyCert : ICertificatePolicy {
         public PolicyCert() {}
         public bool CheckValidationResult(
            ServicePoint sPoint, X509Certificate cert,
            WebRequest wRequest, int certProb) {
            return true;
         }
      }
"@
   [System.Net.ServicePointManager]::CertificatePolicy = new-object PolicyCert
}

# Configuration
$endpoint = "/agents?select=lastKeepAlive&select=id&status=disconnected"
$method = "get"

$protocol = "https"
$host_name = "<WAZUH_SERVER_API_IP>"
```

```
$port = "<WAZUH_SERVER_API_PORT>"
$username = "<WAZUH_API_USER>"
$password = "<WAZUH_API_PASSWORD>"

# Variables
$base_url = $protocol + "://" + $host_name + ":" + $port
$login_url = $base_url + "/security/user/authenticate"
$endpoint_url = $base_url + $endpoint
$base64AuthInfo = [Convert]::ToBase64String([Text.Encoding]::ASCII.GetBytes(("{0}:{1}" -f $username, $passwo
$headers = New-Object "System.Collections.Generic.Dictionary[[String],[String]]"
$headers.Add("Content-Type", 'application/json')
$headers.Add("Authorization", "Basic " + $base64AuthInfo)

Ignore-SelfSignedCerts
$token_response = Invoke-RestMethod -Uri $login_url -Headers $headers
$headers["Authorization"] = "Bearer " + $token_response.data.token

# Request
try{
   $response = Invoke-RestMethod -Method $method -Uri $endpoint_url -Headers $headers
}catch{
   $response = $_.Exception.Response
}

# WORK WITH THE RESPONSE AS YOU LIKE
Write-Output $response.data
```

Aşağıdaki değişkenleri değiştirin:

- `<WAZUH_SERVER_API_IP>`Wazuh sunucunuzun IP adresi ile.
- `<WAZUH_SERVER_API_PORT>`Wazuh sunucusu API port numarasıyla (varsayılan olarak port 5500).
- `<WAZUH_API_USER>`ve `<WAZUH_API_PASSWORD>`doğru belgelerle.

Bağlantısı kesilen aracılar hakkında bilgi almak için Windows uç noktasında PowerShell betiğini çalıştırın:

```
powershell .\get_agent_keep_alive.py
```

Output

```
affected_items                          total_affected_items total_failed_items failed_items
--------------                          -------------------- ------------------ ------------
{@{lastKeepAlive=2020-05-23T12:39:50Z; id=009},  2                    0                {}
@{lastKeepAlive=2020-05-23T12:39:50Z; id=010}}
```

Revision #3
Created 31 December 2024 13:00:16 by Ayşegül Sarıkaya
Updated 31 December 2024 13:26:28 by Ayşegül Sarıkaya