Kaynaklar

Bu bölümde şu anda kullanılabilir olan kaynakların bir listesi ve her birinin kısa bir açıklaması bulunmaktadır.

- /tasks/create/file
- /tasks/create/url
- /tasks/create/submit
- /tasks/list
- /tasks/sample
- /tasks/view
- /tasks/reschedule
- /tasks/delete
- /tasks/report
- /tasks/summary
- /tasks/screenshots
- /tasks/rereport
- /tasks/reboot
- /memory/list
- /memory/get
- /files/view
- /files/get
- /pcap/get
- /machines/list
- /machines/view
- /cuckoo/status
- /vpn/status
- /exit

/tasks/create/file

POST /tasks/create/file

Bir dosyayı bekleyen görevler listesine ekler. Yeni oluşturulan görevin kimliğini döndürür.

Örnek request:

```
curl -H "Authorization: Bearer S4MPL3" -F file=@/path/to/file
http://localhost:8090/tasks/create/file
```

Python kullanılan örnek request:

```
import requests

REST_URL = "<http://localhost:8090/tasks/create/file>"
SAMPLE_FILE = "/path/to/malwr.exe"
HEADERS = {"Authorization": "Bearer S4MPL3"}

with open(SAMPLE_FILE, "rb") as sample:
    files = {"file": ("temp_file_name", sample)}
    r = requests.post(REST_URL, headers=HEADERS, files=files)

# Add your code to error checking for r.status_code.

task_id = r.json()["task_id"]

# Add your code for error checking if task_id is None.
```

Örnek response:

```
{
    "task_id" : 1
}
```

Form parametreleri:

- file (zorunlu) - örnek dosya (multipart kodlu dosya içeriği)
- package (isteğe bağlı) - analiz için kullanılacak analiz paketi
- timeout (isteğe bağlı) (int) - analiz süresi aşımı (saniye cinsinden)
- priority (isteğe bağlı) (int) - göreve atanacak öncelik (1-3)
- options (isteğe bağlı) - analiz paketine iletmek için seçenekler
- machine (isteğe bağlı) - analiz için kullanılacak analiz makinesinin etiketi
- platform (isteğe bağlı) - analiz makinesini seçmek için platform adı (örneğin "windows")
- tags (isteğe bağlı) - makineyi başlatmak için etiketlere göre tanımlama. Bu kullanıldığında
- platformun ayarlanmış olması gerekir. Etiketler virgülle ayrılır
- custom (isteğe bağlı) - analiz ve işleme/bildirme modüllerine geçirilmek üzere özel bir dize
- owner (isteğe bağlı) - birden fazla kullanıcının aynı cuckoo örneğine dosya gönderebileceği durumlarda görev sahibi
- clock (isteğe bağlı) - sanal makine saatinin ayarlanması (biçim %m-%d-%Y %H:%M:%S)
- memory (isteğe bağlı) - analiz makinesinin tam bellek dökümü oluşturmayı etkinleştirme
- unique (isteğe bağlı) - daha önce analiz edilmemiş örnekleri yalnızca gönderme
- enforce_timeout (isteğe bağlı) - tam zaman aşımı değerini zorlamak için etkinleştirme

Durum kodları:

- 200 - hata yok
- 400 - yinelenen dosya tespit edildi (unique seçeneği kullanılıyorsa)

/tasks/create/url

POST /tasks/create/url

Bir dosyayı bekleyen görevler listesine ekler. Yeni oluşturulan görevin kimlik bilgilerini döndürür.

Örnek request:

```
curl -H "Authorization: Bearer S4MPL3" -F url="http://www.malicious.site"
http://localhost:8090/tasks/create/url
```

Python kullanılan örnek request:

```
import requests

REST_URL = "http://localhost:8090/tasks/create/url"
SAMPLE_URL = "http://example.org/malwr.exe"
HEADERS = {"Authorization": "Bearer S4MPL3"}

data = {"url": SAMPLE_URL}
r = requests.post(REST_URL, headers=HEADERS, data=data)

# Add your code to error checking for r.status_code.

task_id = r.json()["task_id"]

# Add your code to error checking if task_id is None.
```

Örnek response:

```
{
   "task_id" : 1
}
```

Form parametreleri:

- url (zorunlu) - analiz edilecek URL (multipart kodlu içerik)

- package (isteğe bağlı) - analiz için kullanılacak paket
- timeout (isteğe bağlı) (int) - analiz süresi (saniye cinsinden)
- priority (isteğe bağlı) (int) - göreve atanacak öncelik (1-3)
- options (isteğe bağlı) - analiz paketine iletilmesi gereken seçenekler
- machine (isteğe bağlı) - analiz için kullanılacak makinenin etiketi
- platform (isteğe bağlı) - analiz makinesini seçmek için platform adı (örneğin "windows")
- tags (isteğe bağlı) - makineyi etiketlere göre başlat. Bu kullanmak için platformun ayarlanmış olması gerekir. Etiketler virgülle ayrılır
- custom (isteğe bağlı) - analiz ve işleme/bildirme modüllerine iletilmek üzere özel bir dize
- owner (isteğe bağlı) - aynı cuckoo örneğine birden fazla kullanıcının dosya gönderebileceği durumda görev sahibi
- memory (isteğe bağlı) - analiz makinesinin tam bellek dökümünün oluşturulmasını etkinleştir
- enforce_timeout (isteğe bağlı) - yürütmenin tam süre boyunca zorlanmasını etkinleştir
- clock (isteğe bağlı) - sanal makine saatinin ayarlanması (format %m-%d-%Y %H:%M:%S)

Durum kodları:

- 200 - hata yok

/tasks/create/submit

POST /tasks/create/submit

Bir veya daha fazla dosyayı ve/veya arşivlere gömülü dosyaları veya yeni oluşturulan görev(ler)in görev ID'lerini içeren bir satırda ayrılmış URL/hash'lerin listesini bekleyen görevlere ekler. Gönderi ID'sini ve yeni oluşturulan görev(ler)in görev ID'lerini döndürür.

Örnek request:

```
# Submit two executables.
curl -H "Authorization: Bearer S4MPL3" http://localhost:8090/tasks/create/submit
-F files=@1.exe -F files=@2.exe

# Submit http://google.com
curl -H "Authorization: Bearer S4MPL3" http://localhost:8090/tasks/create/submit
-F strings=google.com

# Submit http://google.com & http://facebook.com
curl -H "Authorization: Bearer S4MPL3" http://localhost:8090/tasks/create/submit
-F strings=$'google.com\nfacebook.com'
```

Python kullanılan örnek request:

```
import requests

HEADERS = {"Authorization": "Bearer S4MPL3"}

# Submit one or more files.
r = requests.post("http://localhost:8090/tasks/create/submit", files=[
    ("files", open("1.exe", "rb")),
    ("files", open("2.exe", "rb")),
], headers=HEADERS)

# Add your code to error checking for r.status_code.

submit_id = r.json()["submit_id"]
task_ids = r.json()["task_ids"]
errors = r.json()["errors"]
```

```
# Add your code to error checking on "errors".

# Submit one or more URLs or hashes.
urls = [
    "google.com", "facebook.com", "cuckoosandbox.org",
]
r = requests.post(
    "http://localhost:8090/tasks/create/submit",
    headers=HEADERS,
    data={"strings": "\n".join(urls)}
)
```

Örnek response:

```
{
    "submit_id": 1,
    "task_ids": [1, 2],
    "errors": []
}
```

Form parametreleri:

- file (isteğe bağlı) - /tasks/create/file için eski isimle uyumluluk
- files (isteğe bağlı) - kontrol edilecek örnek(ler) ve bekleme sıramıza eklenen örnek(ler)
- strings (isteğe bağlı) - URL'ler ve/veya hash'lerin (VirusTotal API anahtarınızı kullanarak elde edilecek) satırda ayrılmış listesi
- timeout (isteğe bağlı) (int) - analiz süresi sınırlaması (saniye cinsinden)
- priority (isteğe bağlı) (int) - göreve atılacak öncelik (1-3)
- options (isteğe bağlı) - analiz paketi için iletilmesi gereken seçenekler
- tags (isteğe bağlı) - etiketlere göre makine belirtme. Platformun kullanılması gerekiyor. Etiketler virgülle ayrılır
- custom (isteğe bağlı) - analiz ve işleme/bildirime geçirilmek üzere özel dize
- owner (isteğe bağlı) - aynı cuckoo örneğine birden fazla kullanıcının dosya göndermesine izin veriliyorsa görev sahibi
- memory (isteğe bağlı) - analiz makinesinin tam bellek dökümü oluşturmayı etkinleştirme
- enforce_timeout (isteğe bağlı) - yürütmenin tam zaman aşımı değerini zorlamak için etkinleştirme
- clock (isteğe bağlı) - sanal makine saatinin ayarlanması (format %m-%d-%Y %H:%M:%S)

Status kodları:

- 200 - hata yok

/tasks/list

GET /tasks/list/ *(int: limit)* **/** *(int: offset)*

Görevlerin listesini döndürür.

Örnek request:

```
curl -H "Authorization: Bearer S4MPL3" http://localhost:8090/tasks/list
```

Örnek response:


```
{
   "tasks": [
      {
         "category": "url",
         "machine": null,
         "errors": [],
         "target": "http://www.malicious.site",
         "package": null,
         "sample_id": null,
         "guest": {},
         "custom": null,
         "owner": "",
         "priority": 1,
         "platform": null,
         "options": null,
         "status": "pending",
         "enforce_timeout": false,
         "timeout": 0,
         "memory": false,
         "tags": []
         "id": 1,
         "added_on": "2012-12-19 14:18:25",
         "completed_on": null
      },
      {
         "category": "file",
```

```
            "machine": null,
            "errors": [],
            "target": "/tmp/malware.exe",
            "package": null,
            "sample_id": 1,
            "guest": {},
            "custom": null,
            "owner": "",
            "priority": 1,
            "platform": null,
            "options": null,
            "status": "pending",
            "enforce_timeout": false,
            "timeout": 0,
            "memory": false,
            "tags": [
                    "32bit",
                    "acrobat_6",
                ],
            "id": 2,
            "added_on": "2012-12-19 14:18:25",
            "completed_on": null
        }
    ]
}
```


Parametreler:

- limit (isteğe bağlı) (int) - döndürülen görevlerin maksimum sayısı
- offset (isteğe bağlı) (int) - veri ofseti

Durum kodları:

- 200 - hata yok

/tasks/sample

GET /tasks/sample/ *(int: sample_id)*

Belirtilen örnek için görev listesini döndürür.

Örnek request:

```
curl -H "Authorization: Bearer S4MPL3" http://localhost:8090/tasks/sample/1
```

Örnek response:


```
{
   "tasks": [
      {
         "category": "file",
         "machine": null,
         "errors": [],
         "target": "/tmp/malware.exe",
         "package": null,
         "sample_id": 1,
         "guest": {},
         "custom": null,
         "owner": "",
         "priority": 1,
         "platform": null,
         "options": null,
         "status": "pending",
         "enforce_timeout": false,
         "timeout": 0,
         "memory": false,
         "tags": [
                  "32bit",
                  "acrobat_6",
               ],
         "id": 2,
         "added_on": "2012-12-19 14:18:25",
         "completed_on": null
```

```
        }
      ]
    }
```

Parametreler:

- sample_id (gereklidir) (int) - görevleri listelemek için örnek kimliği

Durum kodları:

- 200 - hata yok

/tasks/view

GET /tasks/view/ *(int: id)*

Belirtilen ID'ye sahip görevle ilgili ayrıntıları döndürür.

Örnek request:

```
curl -H "Authorization: Bearer S4MPL3" http://localhost:8090/tasks/view/1
```

Örnek response:


```
{
   "task": {
      "category": "url",
      "machine": null,
      "errors": [],
      "target": "http://www.malicious.site",
      "package": null,
      "sample_id": null,
      "guest": {},
      "custom": null,
      "owner": "",
      "priority": 1,
      "platform": null,
      "options": null,
      "status": "pending",
      "enforce_timeout": false,
      "timeout": 0,
      "memory": false,
      "tags": [
                "32bit",
                "acrobat_6",
            ],
      "id": 1,
      "added_on": "2012-12-19 14:18:25",
      "completed_on": null
   }
```

```
    }
```

Not: Key status için olası değerler:

- pending
- running
- completed
- reported

Parametreler:

- id (gereklidir) (int) - Bakılacak görevin ID'si

Durum kodları:

- 200 - hata yok
- 404 - görev bulunamadı

/tasks/reschedule

GET /tasks/reschedule/ *(int: id)* **/** *(int: priority)*

Belirtilen kimlik ve önceliğe sahip bir görevi yeniden planlayın (varsayılan öncelik 1'dir).

Örnek request:

```
curl -H "Authorization: Bearer S4MPL3" http://localhost:8090/tasks/reschedule/1
```

Örnek response:

```
{
   "status": "OK"
}
```

Parametreler:

- Id (gerekli) (int) - Yeniden planlanması gereken görevin kimliği
- Öncelik (isteğe bağlı) (int) - Görev önceliği

Durum kodları:

- 200 - hata yok
- 404 - görev bulunamadı

/tasks/delete

GET /tasks/delete/ *(int: id)*

Verilen görevi veritabanından kaldırır ve sonuçları siler.

Örnek request:

```
curl -H "Authorization: Bearer S4MPL3" http://localhost:8090/tasks/delete/1
```

Parametreler:

- Id (gerekli) (int) - Silinecek görevin kimliği

Durum kodları:

- 200 - hata yok
- 404 - görev bulunamadı
- 500 - görevi silemiyor

/tasks/report

GET /tasks/report/ *(int: id)* **/** *(str: format)*

Belirtilen görev kimliğiyle ilişkili raporu döndürür.

Örnek request:

```
curl -H "Authorization: Bearer S4MPL3" http://localhost:8090/tasks/report/1
```

Parametreler:

- Id (gerekli) (int) - Raporu almak için görevin kimliği
- format (isteğe bağlı) - [json/html/all/dropped/package_files] dosyasını almak için raporun biçimi. Hiçbiri belirtilmezse JSON raporu döndürülecektir. hepsi tüm sonuç dosyalarını tar.bz2 olarak döndürür, bırakılan dosyaları tar.bz2 olarak düşürür, paket_files dosyaları analiz paketleri tarafından ana bilgisayara yüklenir.

Durum kodları:

- 200 - hata yok
- 400 - geçersiz rapor formatı
- 404 - rapor bulunamadı

/tasks/summary

GET /tasks/summary/ *(int: id)*

Belirtilen görev kimliğiyle ilişkili yoğunlaştırılmış bir raporu JSON biçiminde döndürür.

Örnek request:

```
curl http://localhost:8090/tasks/summary/1
```

Parametreler:

- Id (gerekli) (int) - Raporu almak için görevin kimliği

Durum kodları:

- 200 - hata yok
- 404 - rapor bulunamadı

/tasks/screenshots

GET /tasks/screenshots/ *(int: id)* **/** *(str: number)*

Belirtilen görev kimliğiyle ilişkili bir veya tüm ekran görüntülerini döndürür.

Örnek request:

```
wget http://localhost:8090/tasks/screenshots/1
```

Parametreler:

- Id (gerekli) (int) - Raporu almak için görevin kimliği
- screenshot (isteğe bağlı) - tek bir ekran görüntüsünün sayısal tanımlayıcısı (örn. 0001, 0002)

Durum kodları:

- 404 - dosya veya klasör bulunamadı

/tasks/rereport

GET /tasks/rereport/ *(int: id)*

Belirtilen görev kimliğiyle ilişkili görev için raporlamayı yeniden çalıştırın.

Örnek request:

```
curl -H "Authorization: Bearer S4MPL3" http://localhost:8090/tasks/rereport/1
```

Örnek response:

```
{
   "success": true
}
```

Parametreler:

- Id (zorunlu) (int) - Raporu yeniden çalıştıracak görevin kimliği

Durum kodları:

- 200 - hata yok
- 404 - görev bulunamadı

/tasks/reboot

GET /tasks/reboot/ *(int: id)* **

Mevcut bir analiz kimliğinden veritabanına bir yeniden başlatma görevi ekleyin.

Örnek request:

```
curl -H "Authorization: Bearer S4MPL3" http://localhost:8090/tasks/reboot/1
```

Örnek response:

```
{
   "task_id": 1,
   "reboot_id": 3
}
```

Parametreler:

- Id (gerekli) (int) - Görevin kimliği

Durum kodları:

- 200 - başarılı
- 404 - yeniden başlatma görevi oluşturma hatası

/memory/list

GET /memory/list/ *(int: id)*

Belirtilen görev kimliğiyle ilişkili bellek döküm dosyalarının veya bir bellek döküm dosyasının bir listesini döndürür.

Örnek request:

```
wget http://localhost:8090/memory/list/1
```

Parametreler:

- Id (gerekli) (int) - Raporu almak için görevin kimliği

Durum kodları:

- 404 - dosya veya klasör bulunamadı

/memory/get

GET /memory/get/ *(int: id)* **/** *(str: number)*

Belirtilen görev kimliğiyle ilişkili bir bellek döküm dosyası döndürür.

Örnek request:

```
wget http://localhost:8090/memory/get/1/1908
```

Parametreler:

- Id (gerekli) (int) - Raporu almak için görevin kimliği
- Pid (zorunlu) - tek bir bellek döküm dosyasının sayısal tanımlayıcısı (pid) (örn. 205, 1908)

Durum kodları:

- 404 - dosya veya klasör bulunamadı

/files/view

GET /files/view/md5/ *(str: md5)*

GET /files/view/sha256/ *(str: sha256)*

GET /files/view/id/ *(int: id)*

Belirtilen MD5 hash, SHA256 hash veya kimlikle eşleşen dosyadaki ayrıntıları döndürür.

Örnek request:

```
curl -H "Authorization: Bearer S4MPL3" http://localhost:8090/files/view/id/1
```

Örnek response:


```
{
   "sample": {
      "sha1": "da39a3ee5e6b4b0d3255bfef95601890afd80709",
      "file_type": "empty",
      "file_size": 0,
      "crc32": "00000000",
      "ssdeep": "3::",
      "sha256":
"e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855",
      "sha512":
"cf83e1357eefb8bdf1542850d66d8007d620e4050b5715dc83f4a921d36ce9ce4
7d0d13c5d85f2b0ff8318d2877eec2f63b931bd47417a81a538327af927da3e",
      "id": 1,
      "md5": "d41d8cd98f00b204e9800998ecf8427e"
   }
}
```


Parametreler:

- md5 (isteğe bağlı) - Aranacak dosyanın MD5 hashi
- sha256 (isteğe bağlı) - Aranacak dosyanın SHA256 hashi
- Id (isteğe bağlı) (int) - Aranacak dosyanın kimliği

Durum kodları:

- 200 - hata yok
- 400 - geçersiz arama terimi
- 404 - dosya bulunamadı

/files/get

GET /files/get/ *(str: sha256)*

Belirtilen SHA256 hashi ile eşleşen dosyanın binary içeriğini döndürür.

Örnek request:

> curl -H "Authorization: Bearer S4MPL3"
> http://localhost:8090/files/get/e3b0c44298fc1c149afbf4c8996fb92427ae41e4649b934ca495991b7852b855 > sample.exe

Durum kodları:

- 200 - hata yok
- 404 - dosya bulunamadı

/pcap/get

GET /pcap/get/ *(int: task)*

Verilen görevle ilişkili PCAP içeriğini döndürür.

Örnek request:

```
curl -H "Authorization: Bearer S4MPL3" http://localhost:8090/pcap/get/1 > dump.pcap
```

Durum kodları:

- 200 - hata yok
- 404 - dosya bulunamadı

/machines/list

GET /machines/list

Cuckoo için mevcut olan analiz makinelerinin ayrıntılarını içeren bir liste döndürür.

Örnek request:

```
curl -H "Authorization: Bearer S4MPL3" http://localhost:8090/machines/list
```

Örnek response:


```
{
   "machines": [
      {
         "status": null,
         "locked": false,
         "name": "cuckoo1",
         "resultserver_ip": "192.168.56.1",
         "ip": "192.168.56.101",
         "tags": [
                  "32bit",
                  "acrobat_6",
               ],
         "label": "cuckoo1",
         "locked_changed_on": null,
         "platform": "windows",
         "snapshot": null,
         "interface": null,
         "status_changed_on": null,
         "id": 1,
         "resultserver_port": "2042"
      }
   ]
}
```


Durum kodları:

- 200 - hata yok

/machines/view

GET /machines/view/ *(str: name)*

Verilen adla ilişkili analiz makinesindeki ayrıntıları döndürür.

Örnek request:

```
curl -H "Authorization: Bearer S4MPL3"
http://localhost:8090/machines/view/cuckoo1
```

Örnek response:


```
{
   "machine": {
      "status": null,
      "locked": false,
      "name": "cuckoo1",
      "resultserver_ip": "192.168.56.1",
      "ip": "192.168.56.101",
      "tags": [
                "32bit",
                "acrobat_6",
            ],
      "label": "cuckoo1",
      "locked_changed_on": null,
      "platform": "windows",
      "snapshot": null,
      "interface": null,
      "status_changed_on": null,
      "id": 1,
      "resultserver_port": "2042"
   }
}
```


Durum kodları:

- 200 - hata yok

- 404 - makine bulunamadı

/cuckoo/status

GET /cuckoo/status/

Cuckoo sunucunun durumunu döndürür. 1.3 sürümünde diskspace girişi eklenmiştir. Diskspace girişi, ilgili dizinlerin bulunduğu diskin kullanılan, boş ve toplam disk alanını gösterir. Diskspace girişi, bir Cuckoo düğümünü Cuckoo API aracılığıyla izleme olanağı sağlar. Unutulmamalıdır ki her dizin ayrı ayrı kontrol edilir, çünkü birisi $CUCKOO/storage/analyses için ayrı bir sabit diske bir sembolik bağ oluşturabilir, ancak $CUCKOO/storage/binaries'yi olduğu gibi bırakabilir. (Bu özellik yalnızca Unix altında kullanılabilir!)

1.3 sürümünde cpuload girişi de eklenmiştir - cpuload girişi, sırasıyla son bir dakika, son 5 dakika ve son 15 dakika için CPU yükünü gösterir. (Bu özellik yalnızca Unix altında kullanılabilir!)

Diskspace dizinleri:

- analyses - $CUCKOO/storage/analyses/
- binaries- $CUCKOO/storage/binaries/
- temporary - tmppath as specified in conf/cuckoo.conf

Örnek request:

```
curl -H "Authorization: Bearer S4MPL3" http://localhost:8090/cuckoo/status
```

Örnek response:


```
{
   "tasks": {
      "reported": 165,
      "running": 2,
      "total": 167,
      "completed": 0,
      "pending": 0
   },
   "diskspace": {
      "analyses": {
         "total": 491271233536,
         "free": 71403470848,
         "used": 419867762688
```

```
        },
        "binaries": {
            "total": 491271233536,
            "free": 71403470848,
            "used": 419867762688
        },
        "temporary": {
            "total": 491271233536,
            "free": 71403470848,
            "used": 419867762688
        }
    },
    "version": "1.0",
    "protocol_version": 1,
    "hostname": "Patient0",
    "machines": {
        "available": 4,
        "total": 5
    }
}
```


Durum kodları:

- 200 - hata yok
- 404 - makine bulunamadı

/vpn/status

GET /vpn/status

VPN durumunu döndürür.

Örnek request:

```
curl -H "Authorization: Bearer S4MPL3" http://localhost:8090/vpn/status
```

Durum kodları:

- 200 - hata yok
- 500 - mevcut değil

/exit

GET /exit

Hata ayıklama modundaysa ve werkzeug sunucusunu kullanıyorsa sunucuyu kapatır.

Örnek request:

```
curl -H "Authorization: Bearer S4MPL3" http://localhost:8090/exit
```

Durum kodları:

- 200 - hata yok
- 403 - Bu çağrı sadece hata ayıklama modunda kullanılabilir
- 500 - error