



CHECK POINT

SandBlast™

ZERO-DAY
PROTECTION

THREAT PREVENTION API

JAVIER PADILLA

THREAT PREVENTION REGIONAL EXPERT, MEXICO

jpadilla@checkpoint.com

27/Jul/2017



TOPICS

- What is the API and why is so cool!
- API clients can run everywhere.
- API demonstration.
- Share some customer needs.

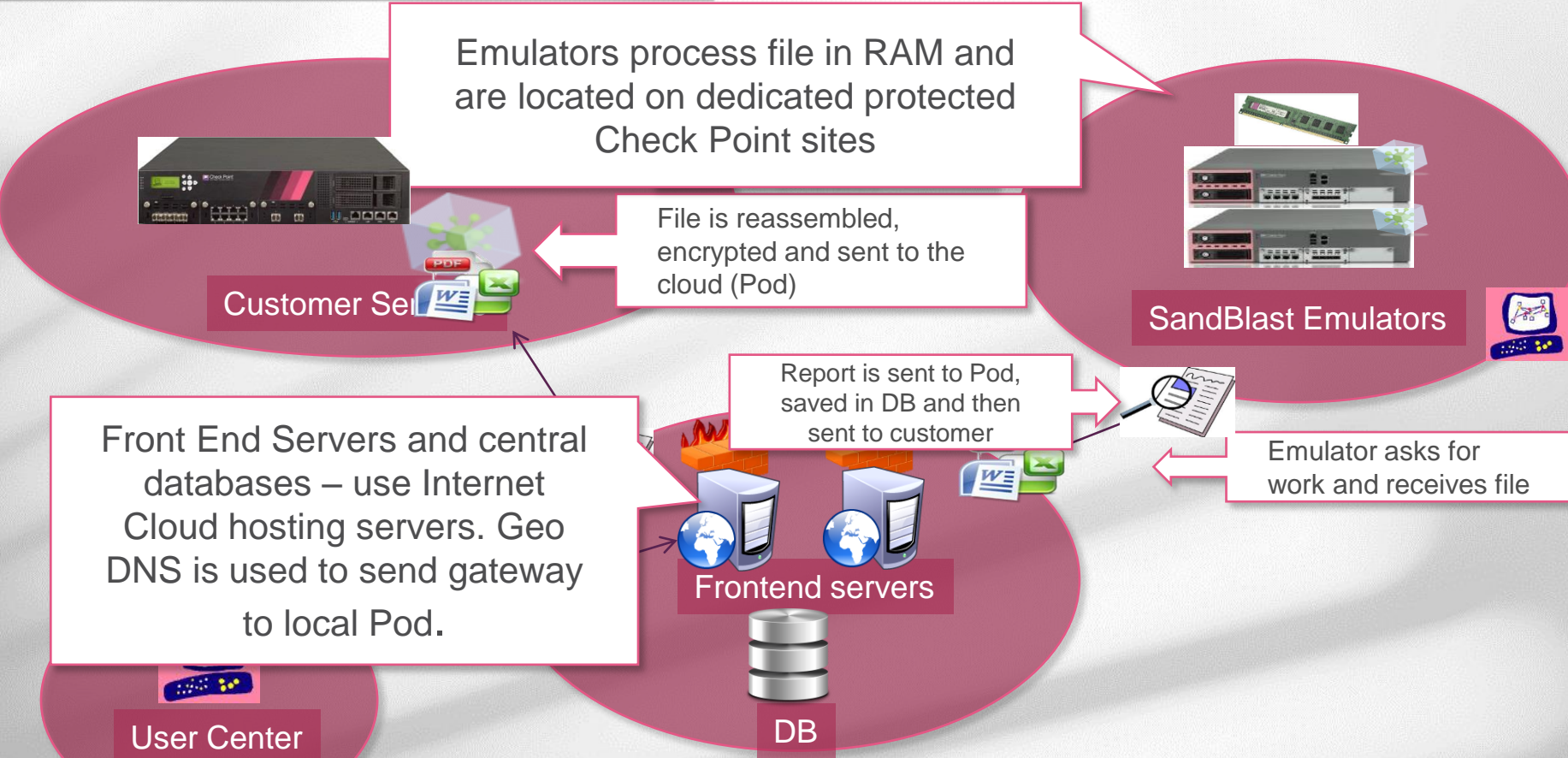


INTRODUCTION

What, Why, where?

- Malware Analysis as a Service

SandBlast Service Overview



Enable API on appliance

- A. **SK 113599**: SandBlast Agent for Browsers - working with Security Gateway or TE Appliance.
 - Enabling the Threat Prevention API
 - Portal Certificate
- B. Latest JHF includes it.
- C. Activate scrubd_webservice using GUI DBEdit



Unified Threat Prevention API

Where?

- Local or Cloud

Threat Prevention API capabilities supported

- AV &TE hash reputation. `['av','te']`
- Submit a file to be analyzed by TE `['te']`
- Get TE early verdict `['te_eb']`
- Choose emulation environments.
- TE Forensics report download.
- Covert a file to PDF `['extraction']`
- Clean specific content from files.

Exclusive on Cloud

- Include Evaluation Key through HTTP headers
- Save Cookies for emulation results to be accurate.



How to Use it

Hands On

- We need some little code

Stuff we need to keep in mind when using the API.

- REST Full API
- JSON
- HTTP Methods
- HTTP Headers
- Base64 Encoding/Decoding
- multipart/form-data
- **Cookies !!!**

JSON

- Java Script Object Notation
 - ✓ Syntax for storing and exchanging data between applications.
 - ✓ Easier to use than XML.
 - ✓ Easy to read.
 - ✓ Very native to handle between different languages.
 - ✓ They work like dictionaries.
 - ✓ It will maintain the type of the variables .

JSON

- Query request to AntiVirus using MD5

```
1 {  
2   "request": [  
3     {  
4       "md5": "8dfa1440953c3d93daafeae4a5daa326",  
5       "features": ["av"]  
6     }  
7   ]  
8 }
```

JSON

- Response from the previous query

```
1 {
2   "response": [
3     {
4       "status": {
5         "code": 1001,
6         "label": "FOUND",
7         "message": "The request has been fully answered."
8       },
9       "md5": "8dfa1440953c3d93daafeae4a5daa326",
10      "file_type": "",
11      "file_name": "",
12      "features": [
13        "av"
14      ],
15      "av": {
16        "malware_info": {
17          "signature_name": "Trojan-Downloader.Win32.Agent.T.kdu",
18          "malware_family": 36471,
19          "malware_type": 87,
20          "severity": 4,
21          "confidence": 5
22        },
23        "status": {
24          "code": 1001,
25          "label": "FOUND",
26          "message": "The request has been fully answered."
27        }
28      }
29    }
30  ]
31 }
```

HTTP METHODS

- HTTP GET- Request information/data.
 - DOWNLOAD (*TE reports / scrubbed files*)
 - QUOTA (*SandBlast **Cloud** Service quota*)
- HTTP POST- Create a resource / Submit data
 - QUERY (*AV & TE hash reputation*)
 - UPLOAD (*Send files to TE & TEX*)

HTTP HEADERS

- Authorization

This is how we submit our API Key, this header is part of **all requests**.

- Content-Type : application/json

The type of data we are sending on this request are **JSON** objects

The screenshot shows a REST client interface with the following details:

- Method: POST
- URL: https://te.checkpoint.com/tecloud/api/v1/file/query
- Params: (empty)
- Active Tab: Headers (2)
- Header 1: Content-Type: application/json
- Header 2: Authorization: TE_API_KEY_9sMr4WvYduaZpBQqvzVp5BPW5RZw1ihw

Method	URL	Params
POST	https://te.checkpoint.com/tecloud/api/v1/file/query	
Authorization	Headers (2)	Body
Pre-request Script	Tests	
<input checked="" type="checkbox"/>	Content-Type	application/json
<input checked="" type="checkbox"/>	Authorization	TE_API_KEY_9sMr4WvYduaZpBQqvzVp5BPW5RZw1ihw

A flexible HTTP Client, some examples

A. Python + requests

B. Curl + Bash

1. Use jq or JSON.sh to read JSON Data

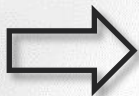
C. Postman

1. <https://www.getpostman.com/collections/517bd6087b29529b17a7>

Malware Analysis proposed flow



Supported
File Format



AV DB /query POST Request
feature: 'av', 'te'
md5 : 23e234242qre
file_name : suspiciousfile.doc



AV DB /query Response
feature: 'av', 'te'
md5 : 23e234242qre
UNKNOWN, FOUND

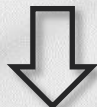


TE /upload POST Request
feature: 'te'
md5: 23e234242qre
Images: <image_uuid>,<rev>
report: pdf,xml



status: code: 1004
feature: 'av', 'te'
label: NOT_FOUND
IT IS UNKNOWN, UPLOAD IT

status: code: 1001
feature: 'av'
label: FOUND
THIS IS A KNOWN MALWARE



status: code: 1003
feature: 'te',
label: PENDING



status: code: 1001
feature: 'te', label: FOUND
pdf_report: UUID
verdict: malicious / benign



TE Status /query POST
feature: 'te'
md5 : 23e234242qre
file_name : suspiciousfile.doc



/download GET Request
Params : < report_UUID >
Get forensics report



status: code: 1001
feature: 'te'
label: FOUND
pdf_report: UUID
Verdict: malicious / benign



QUERY API

What do you know about
this file?

- POST Method

QUERY API

1. API uses POST Method
2. Multiple hashes are supported (SHA1 & MD5*)
3. API Endpoint is <https://te.checkpoint.com/tecloud/api/v1/file/query>
4. Set Headers:
 1. Content-type : application/json
 2. Authorization : < Evaluation TP API Key>

Example Query just AV

POST Params

Authorization Headers (2) Body Pre-request Script Tests [Code](#)

<input checked="" type="checkbox"/>	Content-Type	application/json	<input type="button" value="⋮"/>	<input type="button" value="✕"/>	<input type="button" value="Bulk Edit"/>	<input type="button" value="Presets"/>
<input checked="" type="checkbox"/>	Authorization	TE_API_KEY_WbcFFFdp2BDxbResvlbs9sSaMgoR4Gwbal	<input type="button" value="⋮"/>	<input type="button" value="✕"/>		
	key	value				

Is this hash known to be malicious?



```
1 import requests
2 import json
3
4 service = "https://172.20.22.220:18194/tecloud/api/v1/file/"
5 request = { "request": [
6     { "md5": "764bfd2b86c9eb5ad1432514c8802a10",
7       "features": ["av","te"] ]}]
8
9 response = requests.post(service + 'query', data = json.dumps(request), verify=False)
10 responseText = json.loads(response.text)
```

QUERY API request for AntiVirus only

Query ar

{ "request":

```
1  {
2    "response": [
3      {
4        "status": {
5          "code": 1001,
6          "label": "FOUND",
7          "message": "The request has been fully answered."
8        },
9        "md5": "7234f226dd368a5e2caa7998a758d418",
10       "file_type": "",
11       "file_name": "",
12       "features": [
13         "av"
14       ],
15       "av": {
16         "malware_info": {
17           "signature_name": "Backdoor.Win32.Androm.lvbh.a",
18           "malware_family": 236921,
19           "malware_type": 98,
20           "severity": 4,
21           "confidence": 5
22         },
23         "status": {
24           "code": 1001,
25           "label": "FOUND",
26           "message": "The request has been fully answered."
27         }
28       }
29     ]
30   }
31 }
```

QUERY API request for AntiVirus only

Status of the FULL request.



```
1 {  
2   "response": [  
3     {  
4       "status": {  
5         "code": 1001,  
6         "label": "FOUND",  
7         "message": "The request has been fully answered."  
8       },  
9       "md5": "7234f226dd368a5e2caa7998a758d418",  
10      "file_type": "",  
11      "file_name": "",  
12      "features": [  
13        "av"  
14      ],  
15      "av": {  
16        "malware_info": {  
17          "signature_name": "Backdoor.Win32.Androm.lvbh.a",  
18          "malware_family": 236921,  
19          "malware_type": 98,  
20          "severity": 4,  
21          "confidence": 5  
22        },  
23        "status": {  
24          "code": 1001,  
25          "label": "FOUND",  
26          "message": "The request has been fully answered."  
27        }  
28      }  
29    ]  
30  }  
31 }
```

Antivirus Response details



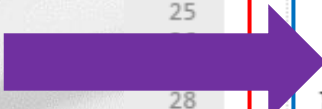
```
16      "av": {  
17        "malware_info": {  
18          "signature_name": "Backdoor.Win32.Androm.lvbh.a",  
19          "malware_family": 236921,  
20          "malware_type": 98,  
21          "severity": 4,  
22          "confidence": 5  
23        }  
24      }
```

Malware info for the specified Hash



```
23        "status": {  
24          "code": 1001,  
25          "label": "FOUND",  
26          "message": "The request has been fully answered."  
27        }  
28      }
```

Status of the AV request



```
23        "status": {  
24          "code": 1001,  
25          "label": "FOUND",  
26          "message": "The request has been fully answered."  
27        }  
28      }
```



UPLOAD API

What could you know
about this file?

- POST Method

UPLOAD API

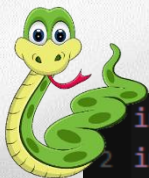
1. API uses POST Method
2. API Endpoint is <https://te.checkpoint.com/tecloud/api/v1/file/upload>
3. Set Headers:
 1. Content-type : application/json
 2. Authorization : < Evaluation TP API Key >
4. **Files** and requests encoded as multipart/form-data
 1. <form action="upload" method="post" **enctype="multipart/form-data"**>

Upload to TE

```
1 import requests
2 import json
3
4 service = "https://172.20.22.220:18194/tecloud/api/v1/file/"
5 print 'Uploading API\n'
6
7 with open('mal_files/1fdb378f04b48f77aefde86f38c0473729cdeefa.doc','r') as sample:
8     request = {"request": {"md5": 'xxx',"file_name": 'malware-file.doc',
9                            #"features": ["te"], .. by default
10                           #Selecting no images means emulate with the default set
11                           "te": {"reports": ["pdf","xml"]}},
12                       "images": [{"id": "7e6fe36e-889e-4c25-8704-56378f0830df","revision": 1}, #Win7 32bit
13                                   {"id": "e50e99f3-5963-4573-af9e-e3f4750b55e2","revision": 1}, #WinXP
14                                   {"id": "8d188031-1010-4466-828b-0cd13d4303ff","revision": 1},
15                                   {"id": "5e5de275-a103-4f67-b55b-47532918fa59","revision": 1},
16                                   {"id": "3ff3ddae-e7fd-4969-818c-d5f1a2be336d","revision": 1},
17                                   {"id": "6c453c9b-20f7-471a-956c-3198a868dc92","revision": 1}]]}
18
19 files = {'file':(sample),'request': json.dumps(request)}
20 print '> Request for uploading a file with content-type multipart/form-data'
21 req = requests.post(service + 'upload', files=files , verify=False)
22 print req.text
23 #print req.cookies['te_cookie']
```



Upload to TEX



```
1 import requests
2 import json
3
4 service = "https://te.checkpoint.com/tecloud/api/v1/file/"
5 headers = {'Authorization': 'TE_API_KEY_6M7ooC3diibjLn5Azfpf5na4AKq6PoaSVQNiiWjp'}
6 print 'Uploading API\n'
7
8 with open('mal_files/1fdb378f04b48f77aefde86f38c0473729cdeefa.doc', 'r') as sample:
9     request = {"request": {"md5": 'xxx',
10                          "file_name": 'malware-file.doc',
11                          "features": ["extraction"],
12                          "extraction": {"method": "convert"}
13                      }}
14
15 files = {'file':(sample), 'request': json.dumps(request)}
16 print '> Request for uploading a file with content-type multipart/metadata'
17 req = requests.post(service + 'upload', files=files , headers=headers, verify=False)
18 print req.text
```



DOWNLOAD API

Give me the reports from
the analysis.

- GET Method

DOWNLOAD API

1. API uses GET Method
2. API Endpoint is <https://te.checkpoint.com/tecloud/api/v1/file/download>
3. Set Headers:
 1. Content-type : No content type defined
 2. Authorization : < Evaluation TP API Key >
4. Request parameters are included in the body of the request
 1. `params = {"id":<"report_id"> }`

Download TE Report

```
1 import requests
2 import json
3
4 service = "https://172.20.22.220:18194/tecloud/api/v1/file/"
5
6 print "Fething report "
7 downloadURL = service + "download"
8 params = {"id": "64926B0F-C1AC-D84F-9F5D-A68431A636A9"}
9 req = requests.get( downloadURL, params=params, verify=False)
10     #-Iterate trough req.iter_content to get the report
11 with open ("report.xml","w") as report:
12     report.write(req.content)
13
```





QUOTA API – only for cloud

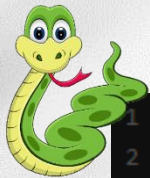
What are my limits?

- GET Method

QUOTA API

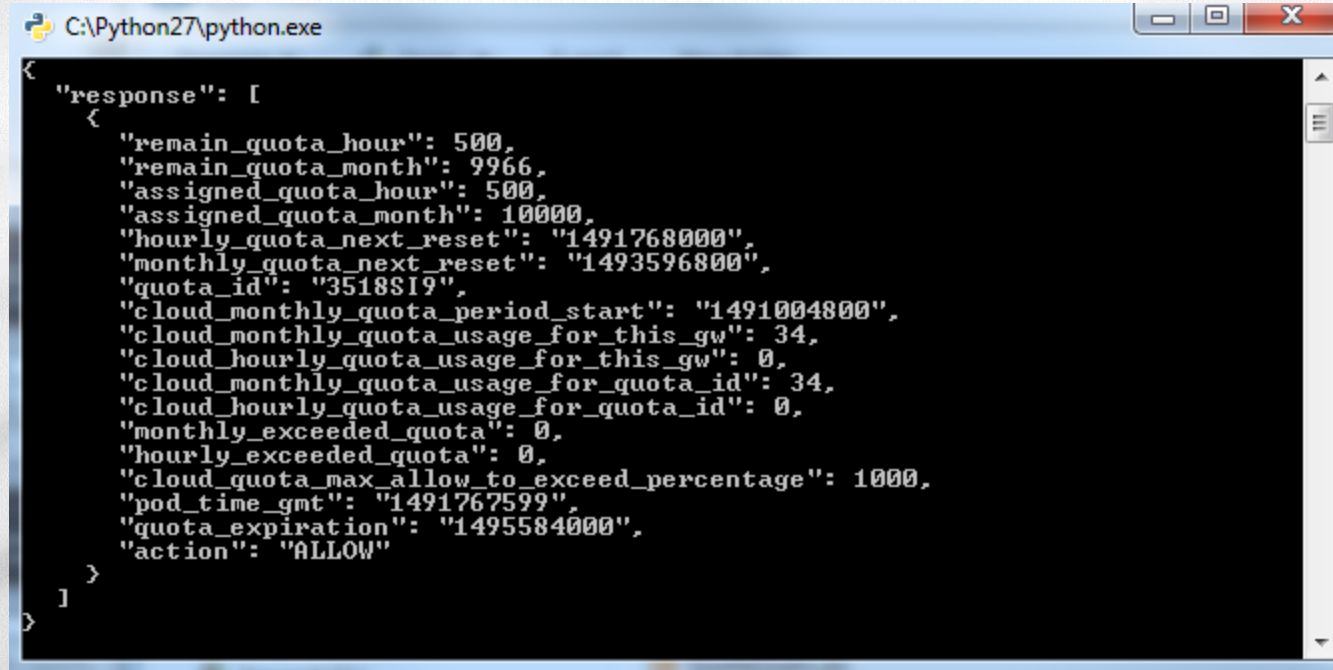
1. API uses GET Method
2. API Endpoint is <https://te.checkpoint.com/tecloud/api/v1/file/quota>
3. Set Headers:
 1. Content-type : No content type defined
 2. Authorization : < Evaluation TP API Key >

What is the quota status?



```
1 import requests
2 import time
3
4 service = "https://te.checkpoint.com/tecloud/api/v1/file/"
5 token = "TE_API_KEY_6M7ooC3diibjLn5Azfpf5na4AKq6PoaSVQNiiWjp"
6 headers = {'Authorization': token}
7 quota = requests.get(service + 'quota', headers = headers, verify=False)
8
9 print quota.text
10 time.sleep(10)
```

Quota is almost new.



```
C:\Python27\python.exe
{
  "response": [
    {
      "remain_quota_hour": 500,
      "remain_quota_month": 9966,
      "assigned_quota_hour": 500,
      "assigned_quota_month": 10000,
      "hourly_quota_next_reset": "1491768000",
      "monthly_quota_next_reset": "1493596800",
      "quota_id": "3518SI9",
      "cloud_monthly_quota_period_start": "1491004800",
      "cloud_monthly_quota_usage_for_this_gw": 34,
      "cloud_hourly_quota_usage_for_this_gw": 0,
      "cloud_monthly_quota_usage_for_quota_id": 34,
      "cloud_hourly_quota_usage_for_quota_id": 0,
      "monthly_exceeded_quota": 0,
      "hourly_exceeded_quota": 0,
      "cloud_quota_max_allow_to_exceed_percentage": 1000,
      "pod_time_gmt": "1491767599",
      "quota_expiration": "1495584000",
      "action": "ALLOW"
    }
  ]
}
```


Code Resources

Threat Prevention API reference guide

Threat Prevention API Python Client

<https://bitbucket.org/chkp/teapi/overview>

SandBlast upload portal (NodeJS) by Martin Koldovsky

<https://github.com/mkol5222/sbport>

PooMA (Python)

<https://bitbucket.org/devilbsd/pooma>

Minimal requests (Python)

<https://bitbucket.org/devilbsd/tpapi/>

One File Portal NodeJs based

The screenshot displays the SandBlast test portal interface. At the top, there is a navigation bar with links for 'SandBlast Portal', 'SBA Forensics', 'Threat Map', 'Threat Portal', and 'Threat Wiki'. Below the navigation bar is a dashed box containing the text 'Drop files or click to upload'. The main content area shows a file upload result for 'Order samples.gz'. The file is identified as 'application/gzip' and is 1.41 MB in size. It was found on 'Mon Apr 11 2016 10:13:00 GMT+0200 (Central Europe Daylight Time)'. The file's origin is 'file found Thu Mar 23 2017 21:27:10 GMT+0100 (Central Europe Standard Time)'. The file's MD5 hash is '21e8eccb4aef18c3fd27d4802d4e42b' and its SHA1 hash is '9d2cf0101f90a3fd73f117a9e2674f87ad603c21'. The verdict is 'malicious', indicated by a red 'Malicious' label. Below the verdict, there are two image thumbnails. At the bottom, a detailed JSON report is displayed, showing the file's status, hashes, features, and a full report with a verdict of 'malicious' and a full report ID of '375e84a6-3521-4e82-bc82-eeee7c7b03cd'.

```
Object {status: Object, sha1: "9d2cf0101f90a3fd73f117a9e2674f87ad603c21",
md5: "21e8eccb4aef18c3fd27d4802d4e42b", file_type: "gz", file_name: "Order samples.gz"...}
  * status: Object
    sha1: "9d2cf0101f90a3fd73f117a9e2674f87ad603c21"
    md5: "21e8eccb4aef18c3fd27d4802d4e42b"
    file_type: "gz"
    file_name: "Order samples.gz"
  * features: Array[2]
    * te: Object
      combined_verdict: "malicious"
      severity: 4
      confidence: 3
      trust: 10
    * images: Array[2]
      * @: Object
        * report: Object
          verdict: "malicious"
          full_report: "375e84a6-3521-4e82-bc82-eeee7c7b03cd"
          pdf_report: "2f3bfe26-3246-4883-bab4-108d7ec4d6c3"
          xml_report: "581977c-7cba-4973-b1d7-2d8a2e444151"
```



In resume

- Check Point's Threat Prevention Unified API is indeed very cool.
- API clients can run everywhere.
- Know about your tools and resources for API projects.

Thank You.