SYHUNT HYBRID: WEB API

The information in this document applies to **version 6.9.8.2** of Syhunt Hybrid.

INTRODUCTION

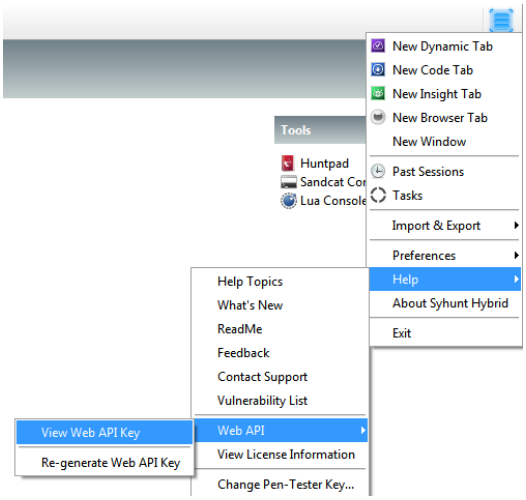
Syhunt Hybrid Platinum comes with a simple, easy-to-use web API that allows to launch dynamic and source code scans, and get status, report and log of a launched scan session. The API expects POST requests with a JSON body and responds with a JSON body. When getting a scan report exceptionally, the response body will have XML or JSON format.

BEFORE STARTING

Make sure the Syhunt API server is up and running:

On Linux, use the command **ScanCore -apisignal:start** to start the API server and **ScanCore -apikeygen** to generate a valid web API key.

On Windows, go to the directory where Syhunt is installed (usually C:\Program Files\Syhunt Hybrid\), enter the Server subdirectory and launch the nginx server by double-clicking the nginx executable. Obtain a web API key simply by launching the Syhunt Hybrid software and going to the View Web API Key help menu option (as shown in the screenshot below).



Visit http://localhost:8017/ and make sure you see a default page for Nginx. The web API is available through **hostname:8017/syhunt/launch.lua** and **/syhunt/results.lua**. Currently, all requests sent to these scripts must contain a valid web API key as part of the body.

LAUNCH A DYNAMIC SCAN

POST /syhunt/launch.lua

Body Type: raw, JSON *(as shown below)*

```
{
  sessionname: "Test",
  starturl: "http://127.0.0.1",
  huntmethod: "appscan",
  reporttemplate: "Complete",
  apikey: "YOUR_API_KEY"
}
```

Keys explained:

- `sessionname` (optional) - must contain an unique, alphanumeric session name. If omitted, a random one will be generated.
- `starturl` (required) - the target URL
- `huntmethod` (optional) - a valid scan method name. For a list of valid hunt methods, see Differences between hunt methods.
- `reporttemplate` (optional) - a valid template report name (Standard, Comparison, Compliance or Complete)
- `reportgl` (optional) - If true, returns a JSON output using GitLab format.
- `timelimit` (optional) - Sets the maximum scan time limit (default: no limit). If the time is reached, the scan is aborted. Examples: 1d, 3h, 2h30m, 50m
- `apikey` (required) - Your Syhunt Web API Key (see the introduction section)

Response body:


```
{
   "huntmethod": "appscan",
   "pid": 5108,
   "result": true,
   "resultstr": "",
   "sessionname": "Test",
   "sessiontype": "dynamic"
}
```


Keys explained:

- `pid` - An unique process ID associated with the active scan,
- `result` - A true result means the scan has been launched. Otherwise a false means it was not possible to start the scan.
- `resultstr` - If the result was false, this will contain an error description.
- `sessionname` - The session name associated with the scan. If you omitted the sessionname key within the request body, this key will contain a randomly generated session name. You must use this session name as part of subsequent requests when getting scan results.
- `sessiontype` - The type of the session (dynamic or code scan)

LAUNCH A SOURCE CODE SCAN

POST /syhunt/launch.lua

Body Type: raw, JSON *(as shown below)*

```
{
  sessionname: "Test",
  sourcetarget: "P:\\Private\\MyWebApp\\",
  huntmethod: "normal",
  reporttemplate: "Complete",
  apikey: "YOUR_API_KEY"
}
```

Keys explained:

- `sessionname` (optional) - must contain an unique, alphanumeric session name. If omitted, a random one will be generated.
- `sourcetarget` (required) - a local target directory or a GIT URL
- `huntmethod` (optional) - a valid scan method name. For a list of valid hunt methods, see Differences between hunt methods.
- `reporttemplate` (optional) - a valid template report name (Standard, Comparison, Compliance or Complete)
- `reportgl` (optional) - If true, returns a JSON output using GitLab format.
- `timelimit` (optional) - Sets the maximum scan time limit (default: no limit). If the time is reached, the scan is aborted. Examples: 1d, 3h, 2h30m, 50m
- `apikey` (required) - Your Syhunt Web API Key (see the introduction section)

Response body:

```
{
   "huntmethod": "normal",
   "pid": 4476,
   "result": true,
   "resultstr": "",
   "sessionname": "Test2",
   "sessiontype": "code"
}
```


Keys explained:

- `pid` - An unique process ID associated with the active scan,
- `result` - A true result means the scan has been launched. Otherwise a false means it was not possible to start the scan.
- `resultstr` - If the result was false, this will contain an error description.
- `sessionname` - The session name associated with the scan. If you omitted the sessionname key within the request body, this key will contain a randomly generated session name. You must use this session name as part of subsequent requests when getting scan results.
- `sessiontype` - The type of the session (dynamic or code scan)

LAUNCH A SOURCE CODE SCAN (PROJECT URL)

POST /syhunt/launch.lua

Body Type: raw, JSON *(as shown below)*

```
{
  sessionname: "Test",
  sourcetarget: "https://github.com/drnic/php-helloworld.git",
  sourcebranch: "master",
  huntmethod: "normal",
  reporttemplate: "Complete",
  apikey: "YOUR_API_KEY"
}
```

GET RESULTS (STATUS, REPORT OR LOG)

GET STATUS

POST /syhunt/results.lua

Body Type: raw, JSON *(as shown below)*

```
{
  sessionname: "Test",
  resulttype: "status",
  apikey: "YOUR_API_KEY"
}
```

Keys explained:

- `sessionname` (required) - the name of a scan session you want to obtain results
- `resulttype` (required) - the type of result you expect (can be status, report_xml or session_log)

Response body:

```
{
   "report_xml_available": true,
   "report_json_available": true,
   "session_log_available": true,
   "sessionname": "Test",
   "status": "Completed"
}
```

Keys explained:

- `report_xml_available` - True if a XML report is already available, false otherwise.
- `report_json_available` - True if a JSON report is already available, false otherwise.
- `session_log_available` - True if a scan log is already available, false otherwise.
- `status` - the scan session status (Scanning or Completed, when it has finished)

GET REPORT

POST /syhunt/results.lua

Body Type: raw, JSON *(as shown below)*

```
{
  sessionname: "Test",
  resulttype: "report_xml",
  apikey: "YOUR_API_KEY"
}

// Alternatively you can obtain the JSON report
{
  sessionname: "Test",
  resulttype: "report_json",
  apikey: "YOUR_API_KEY"
}
```

If report_xml is used, the response body will contain the scan report in XML format, as shown in the example below.

```
<?xml version="1.0"?>
<report>
   <report_title>Syhunt Scanner Report</report_title>

   (...)

   <scanner_version>6.5.0.0</scanner_version>
</report>
```

GET LOG

POST /syhunt/results.lua

Body Type: raw, JSON *(as shown below)*

```
{
  sessionname: "Test",
  resulttype: "session_log",
  apikey: "YOUR_API_KEY"
}
```

The response body will contain the scan log in text format.