Flawnter API Security Testing Guide

The Flawnter API security testing feature will scan your application REST APIs for security vulnerabilities. Currently it supports Postman collections v2.1 in JSON format. If you have OpenAPI Specification (OAS) file, you can import in Postman as collection and then export it to JSON file. You can also use any 3rd party tool that can convert from OpenAPI to Postman collection. Our API security testing is similar to **Dynamic Application Security Testing** (DAST) except it tests only the APIs you define in the JSON file. Note when *deepscan* flag in flawnter.cfg is set to false the API security scan will perform faster scans, but if you want more deeper and accurate scans set the flag to true.

Note: All the authenticated keys/tokens/passwords/etc. must be test accounts. We also recommend extending the authentication token expiration to at least an hour or more to give enough time for the scanner to complete the test.

Authenticated Scan using API Key

To use API key you can just simply add header with name apikey or anything you want. Currently Flawnter supports API keys in the request header only.

Example request using apikey in the header name:

```
"header": [
                {
                  "key": "apikey",
                  "type": "text",
                  "value": "d4638ae73e774585b0357d9de2e3f6b5"
                  }
   ]
```

Authenticated Scan using Bearer Token

To use bearer token you can add Authorization header and set the token as the value.
OAuth2 access token can also be used as bearer token.

Example request using bearer token in the Authorization header:

```
"header": [
            {
              "key": " Authorization",
              "type": "text",
              "value": "Bearer d4638ae73e774585b0357d9de2e3f6b5"
              }
   ]
```

Authenticated Scan using Basic Auth

To use basic auth you can add Authorization header and set the auth token in format username:password and encode it using base64. For Digest authentication use the Digest values instead of Basic. In a Digest authentication the hash of the password is sent to the server.

Example request using basic auth in the Authorization header. Notice the value is in base64.

```
"header": [
            {
              "key": " Authorization",
              "type": "text",
              "value": "Basic dXNlcjE6bXlwYXNz"
              }
   ]
```

Authenticated Scan using JWT Token

To use JWT token you can add Authorization header and set the value to JWT token. The token has 3 parts. The header, the payload and the signature at the end. All the parts are encoded in base64 and separated by dot (.). You can generate the JWT token using Postman Authorization JWT Bearer type. You can also use any other tools and code to generate the JWT token. OAuth2 access token can also be used as JWT token.

Example request using JWT token:

```
"header": [
                 {
                   "key": " Authorization",
                   "type": "text",
                   "value": "Bearer eyJhbGciOiJIUzI1NiJ9.eyJuYW1lIjoiam9lIn0.TmYhceh-
AkfUGgiebNAN8UwbLfRk2IS-STbtkwrq1g0"
                   }
    ]
```

Authenticated Scan using Cookie Session ID

To use Cookie session id, add the session id in the header Cookie. Normally the Cookie session id is set by the server side application after it validates the username and password received from the client side code. In order capture the session id you can use Postman or other tools/proxies. For web applications if it’s using session id, you can retrieve it from the browser developer tools cookies section.

Example request using session id in the Cookie header:

```
"header": [

                   {

                             "key": "Cookie",
                             "type": "text",
                             "value": "PHPSESSID=vo8fqdf84rk7ih6k2ve6sg3m8kmruj4le4ikb6af"

                   }

],
```

Variables in Postman Collection

If you have variables (in format {{variable}}) in your Postman collection, make sure you have defined the variables in “variable” JSON array. It should be same level as “item” JSON array. If you have no variables defined then just simply find and replace the {{variable}} with actual value.

Example of using variable hostName:

```
{
        "info": {
                "_postman_id": "1a48cb1c-03f5-4522-bc98-ad7fdfb2a189",
                "name": "Example API",
                "description": "Testing",
                "schema": "https://schema.getpostman.com/json/collection/v2.1.0/collection.json"
        },
        "item": [
                {
                "url": "https://{{hostName}}/getname"
                }
     ],
     "variable": [
                {
                "key": "hostName",
                "value": "api.example-domain.com",
                "type": "string"
                }
        ]
}
```

Authenticated Logout/Signout Recommendations

If you have defined logout/signout API then make sure the API has **logout** or **signout** in the URL so Flawnter can know those endpoints are to logout from the application.

Sample logout URLs that are supported by Flawnter:
https://exmaple-domain.com/logout
https://exmaple-domain.com/signout
https://exmaple-domain.com/signoff

For sample API json file please download from https://www.flawnter.com/download/samples/api.json.