CloudSEK

How Leaked Twitter API Keys Can be Used to Build a Bot Army

HITBSecConf

CloudSEK

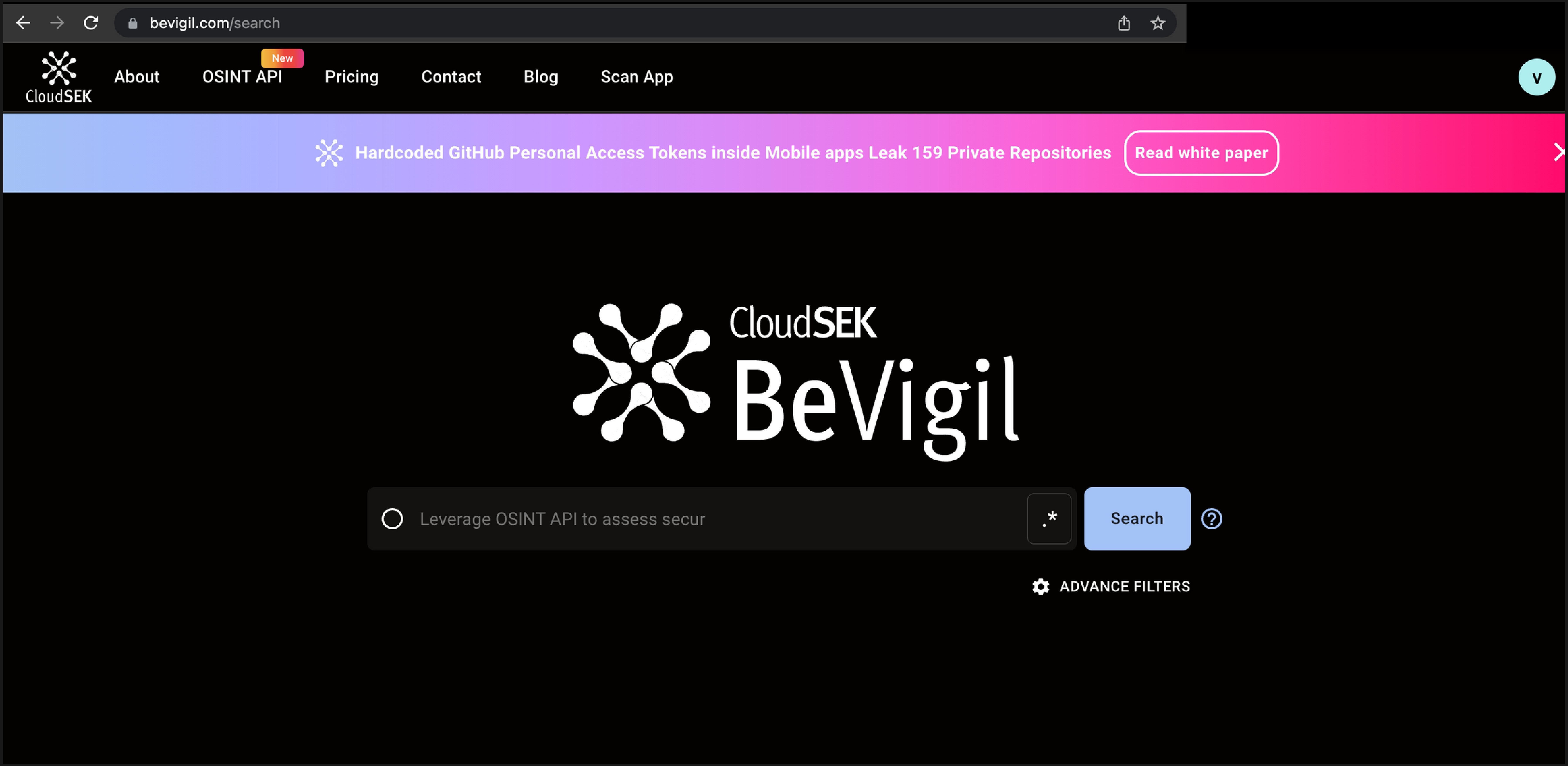
Speaker

Rahul Sasi

Founder, CEO -
CloudSEK

Vishal Singh

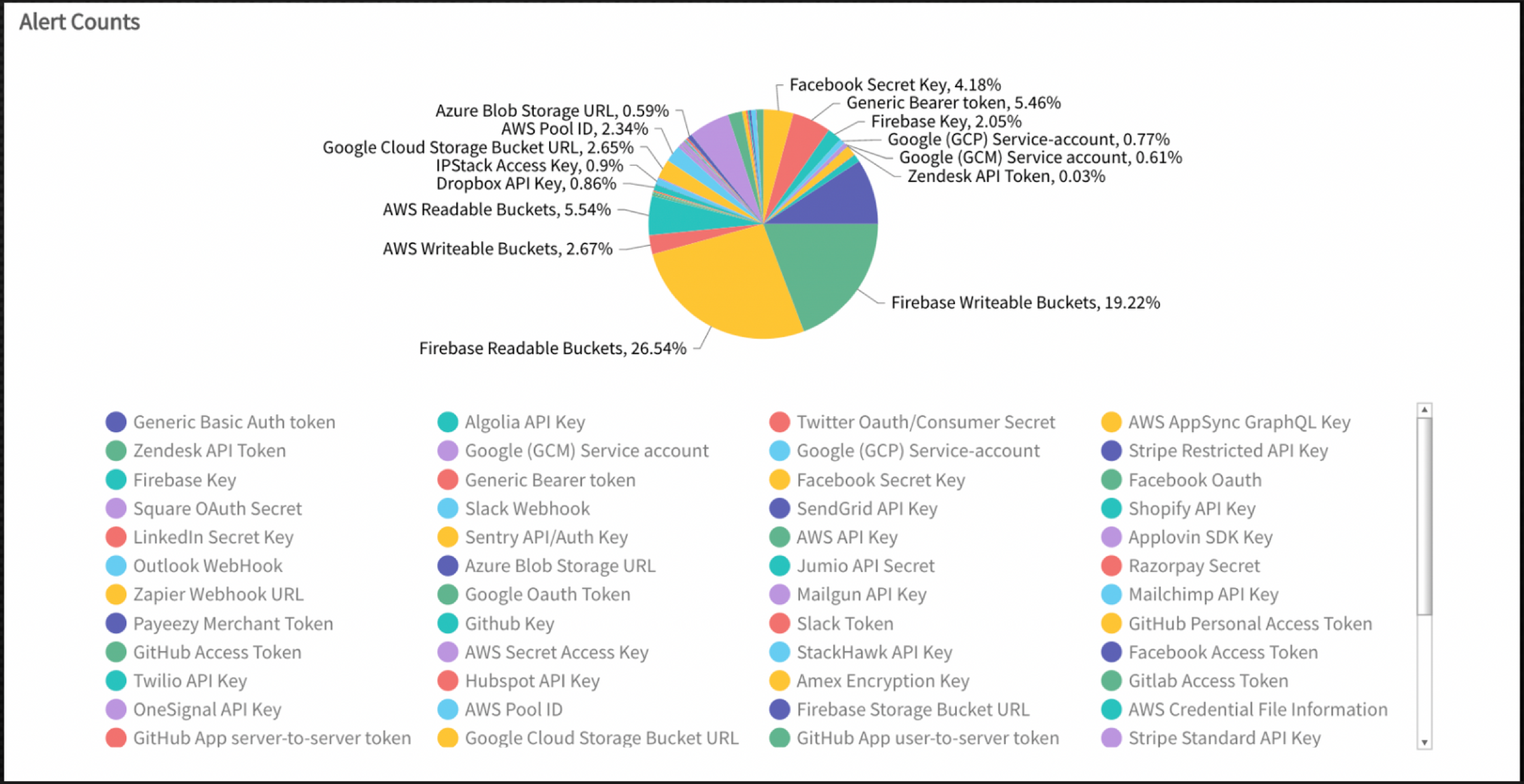
Senior Security Engineer-
CloudSEK



bevigil.com/search
CloudSEK
About
New
OSINT API
Pricing
Contact
Blog
Scan App
V
Hardcoded GitHub Personal Access Tokens inside Mobile apps Leak 159 Private Repositories
Read white paper
CloudSEK
BeVigil
Leverage OSINT API to assess secur
.*
Search
ADVANCE FILTERS

Let's Talk Numbers

1.6 M+ Hardcoded Sensitive Tokens Found at BeVigil



Agenda

01

Key Findings & Source

02

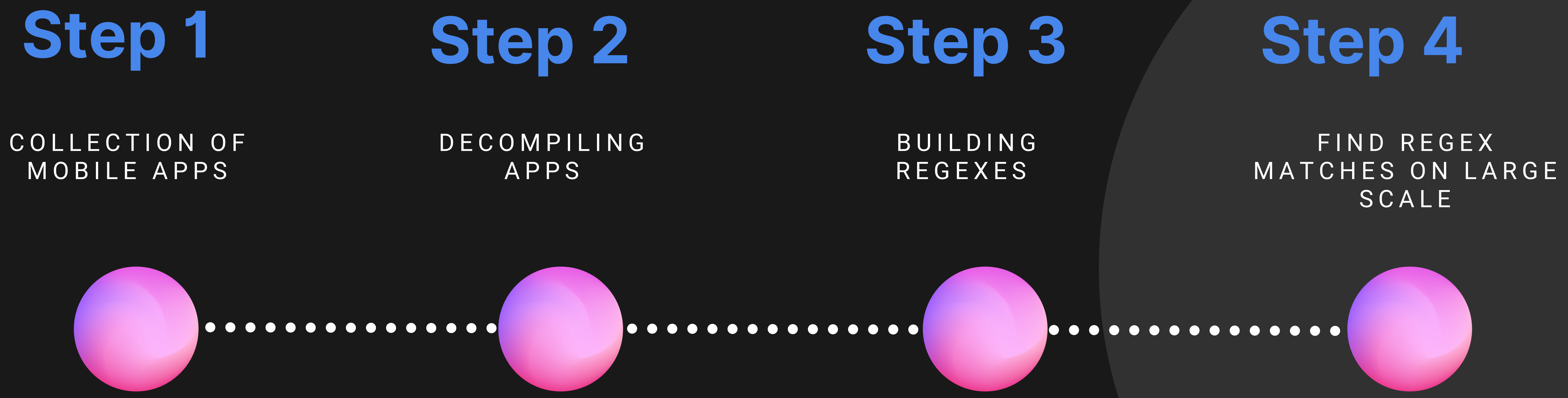
Threats

03

Remediation

KEY FINDING & SOURCE

Our own security search engine



Step 1

COLLECTION OF MOBILE APPS

Step 2

DECOMPILING APPS

Step 3

BUILDING REGEXES

Step 4

FIND REGEX MATCHES ON LARGE SCALE

Collection of Mobile Apps

User submissions

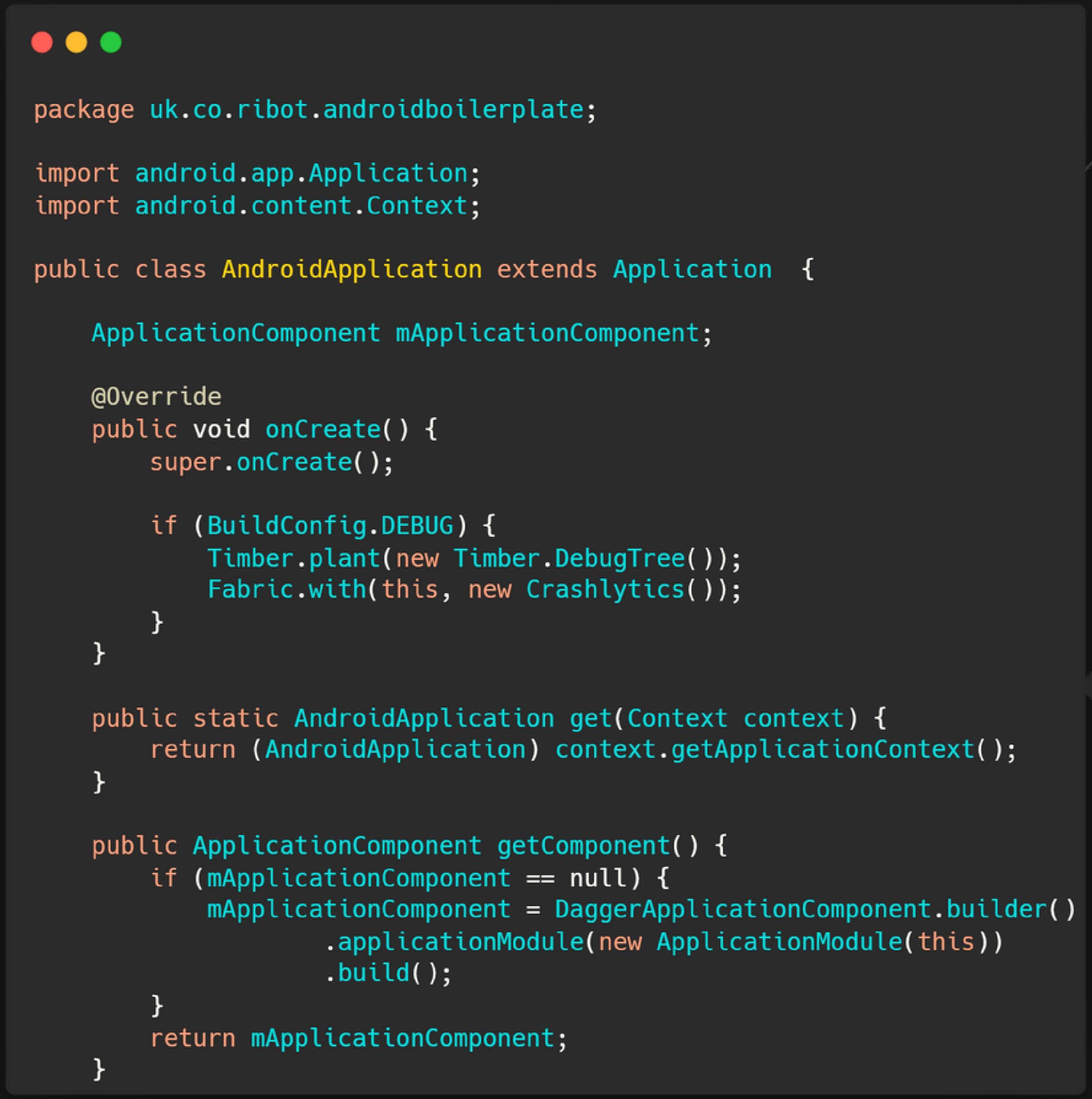
Google Play Store

Decompiling Apps

Open Source Tools such as

- APKTool
- JD-GUI

Open Source Android Decompilers, like JadX



```
package uk.co.ribot.androidboilerplate;

import android.app.Application;
import android.content.Context;

public class AndroidApplication extends Application  {

    ApplicationComponent mApplicationComponent;

    @Override
    public void onCreate() {
        super.onCreate();

        if (BuildConfig.DEBUG) {
            Timber.plant(new Timber.DebugTree());
            Fabric.with(this, new Crashlytics());
        }
    }

    public static AndroidApplication get(Context context) {
        return (AndroidApplication) context.getApplicationContext();
    }

    public ApplicationComponent getComponent() {
        if (mApplicationComponent == null) {
            mApplicationComponent = DaggerApplicationComponent.builder()
                    .applicationModule(new ApplicationModule(this))
                    .build();
        }
        return mApplicationComponent;
    }
```

THE TOUGHEST OF IT ALL

Consumer Key:

```
[tT][wW][iI][tT][tT][eE][rR]([\w\s\-]{0,30})?[":`,=>"\s]{1,5}\b([0-9a-zA-Z]{25})\b["`<"\s]{0,1}
```

Consumer Secret:

```
[tT][wW][iI][tT][tT][eE][rR]([\w\s\-]{0,30})?[":`,=>"\s]{1,5}\b([0-9a-zA-Z]{50})\b["`<"\s]{0,1}
```

Access Token:

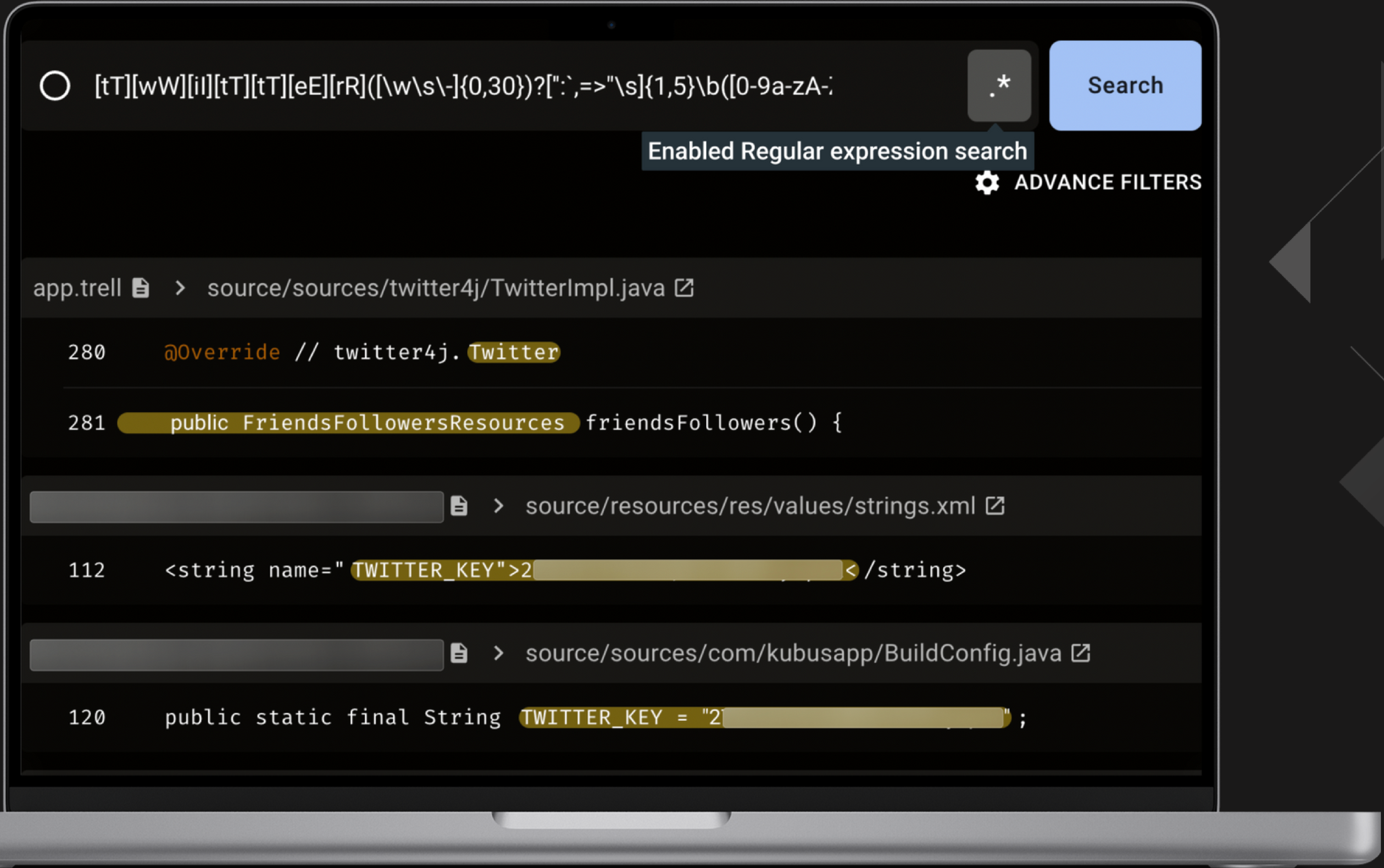
```
[tT][wW][iI][tT][tT][eE][rR]([\w\s\-]{0,30})?[":`,=>"\s]{1,5}\b([0-9]{5,19}\-[0-9a-zA-Z]{30,44})\b["`<"\s]{0,1}
```

Token Secret:

```
[tT][wW][iI][tT][tT][eE][rR]([\w\s\-]{0,30})?[":`,=>"\s]{1,5}\b([0-9a-zA-Z]{45})\b["`<"\s]{0,1}
```

We have build our own RegeEx and grabbed & tested from mulitple sources which detects the Hardcoded keys, tokens, and secrets of the apps.

RegEx Matches on an extensive scale



○ [tT][wW][iI][tT][tT][eE][rR]([\w\s\-]{0,30})?[":`,=>"\s]{1,5}\b([0-9a-zA-

app.trell > source/sources/twitter4j/TwitterImpl.java

```
@Override // twitter4j.Twitter
public FriendsFollowersResources friendsFollowers() {
```

[redacted] > source/resources/res/values/strings.xml

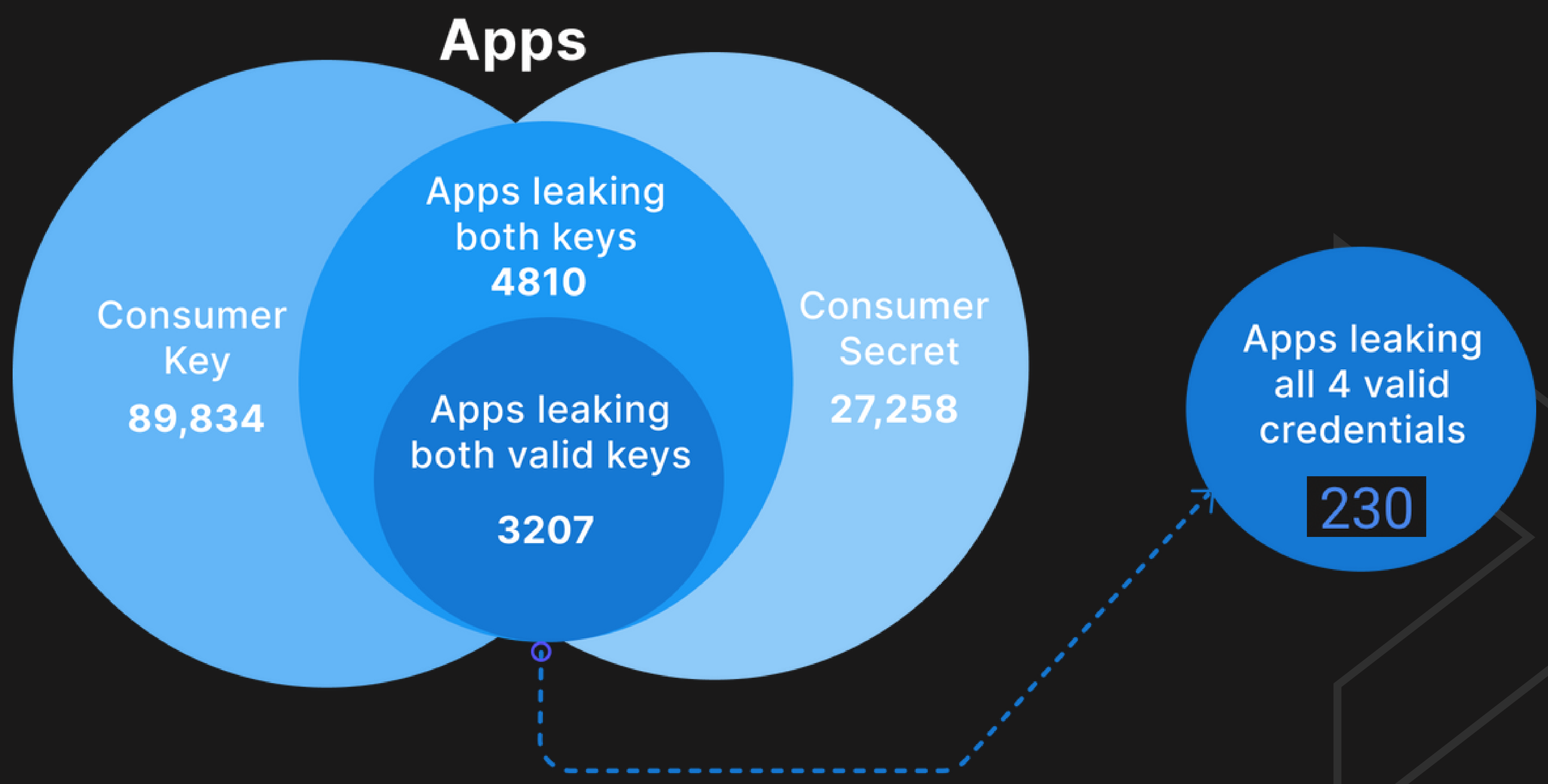
```
<string name="TWITTER_KEY">2[redacted]</string>
```

[redacted] > source/sources/com/kubusapp/BuildConfig.java

```
public static final String TWITTER_KEY = "2[redacted]";
```

Data Analysis

Uncovered **3207 apps**, leaking **Twitter API keys**, that can be utilized to **gain access** to or to **take over** Twitter accounts.


Apps
Consumer Key 89,834
Apps leaking both keys 4810
Apps leaking both valid keys 3207
Consumer Secret 27,258
Apps leaking all 4 valid credentials 230

THREATS

Authentication

The Twitter API uses access controls such as:

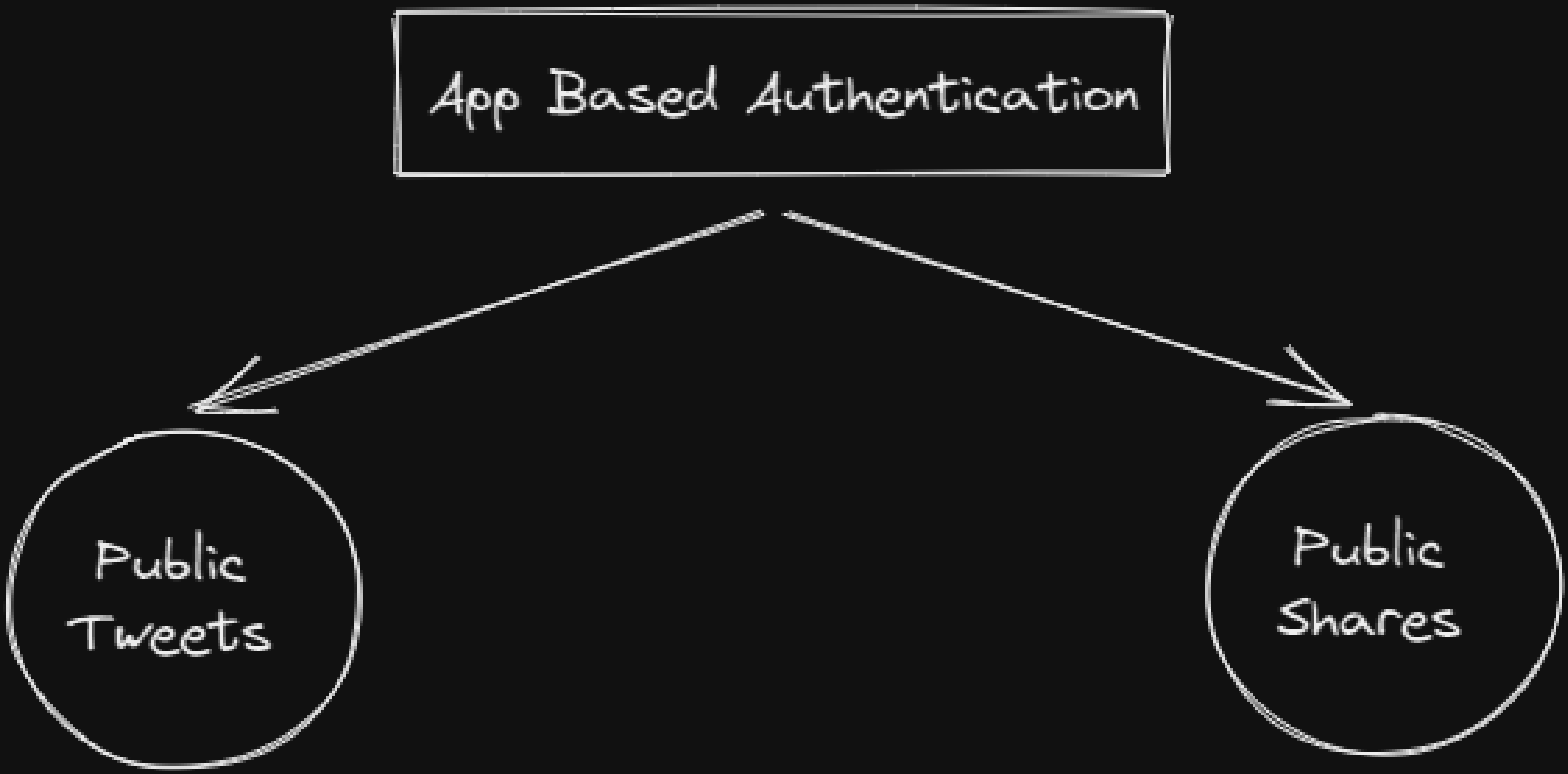
App-Based Authentication

Not Tied to an User Session

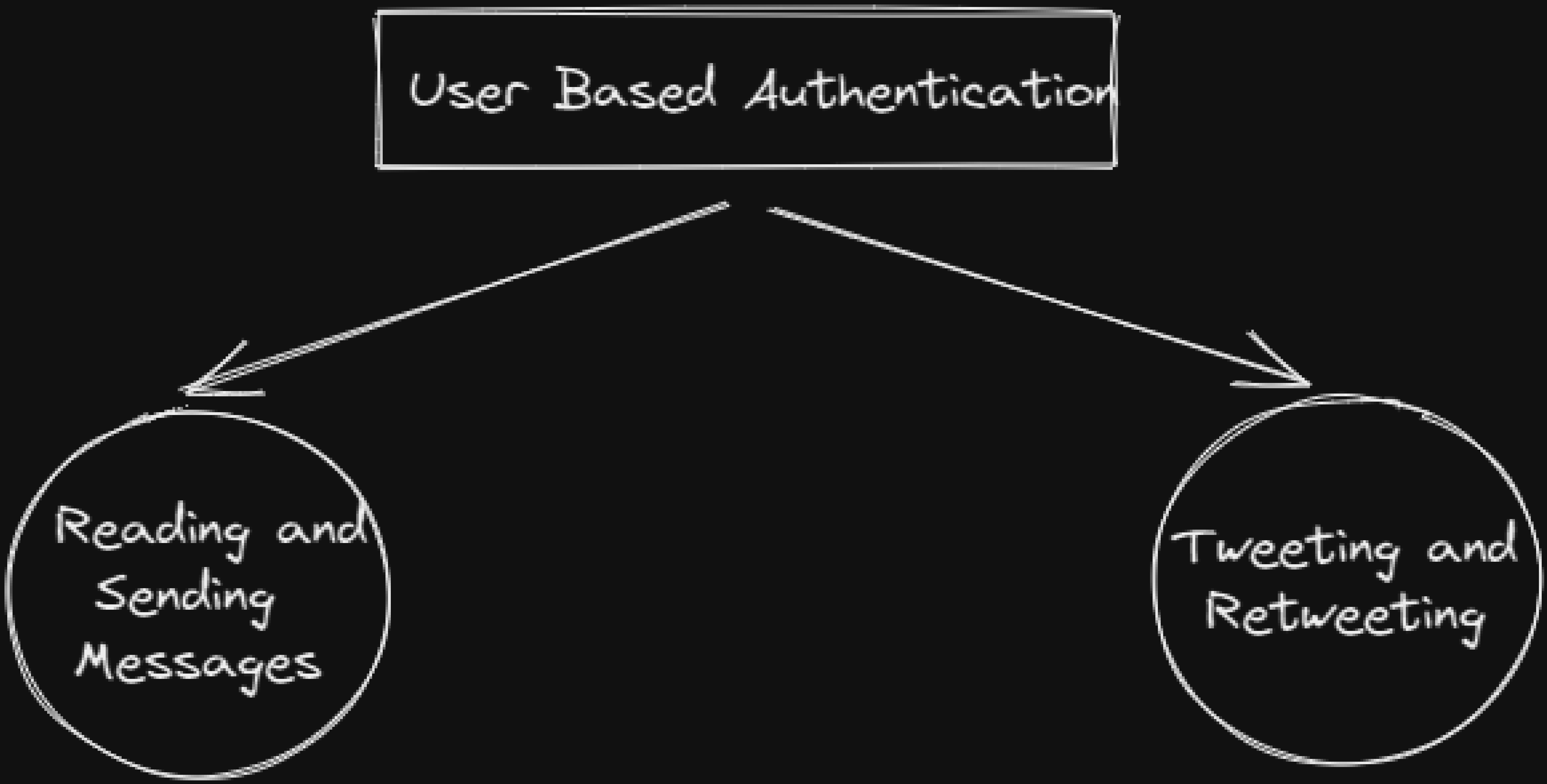
User-Based Authentication

Tied to an User Session

For this, an OAuth 2.0 Bearer Token is used. This can be obtained by passing the API Key and Secret through the POST oauth2/token endpoint. Only 2 keys are required.



For this, the OAuth 1.0a authentication mechanism is used. This requires an Access Token combined with Access Secret . All 4 Keys are Required for this Authentication.



Naming Convention For Keys

Client Credentials

Key

Alternate name used would be

API Key

Consumer API Key

Consumer Key

Customer Key

oauth_consumer_key

Secret

Alternate name used would be

App Key Secret

API Secret Key

Consumer Secret

Consumer Key secret

Customer Key secret

oauth_consumer_secret

Token Credentials

Token

Alternate name used would be

Access token

Token

resulting oauth_token

Secret

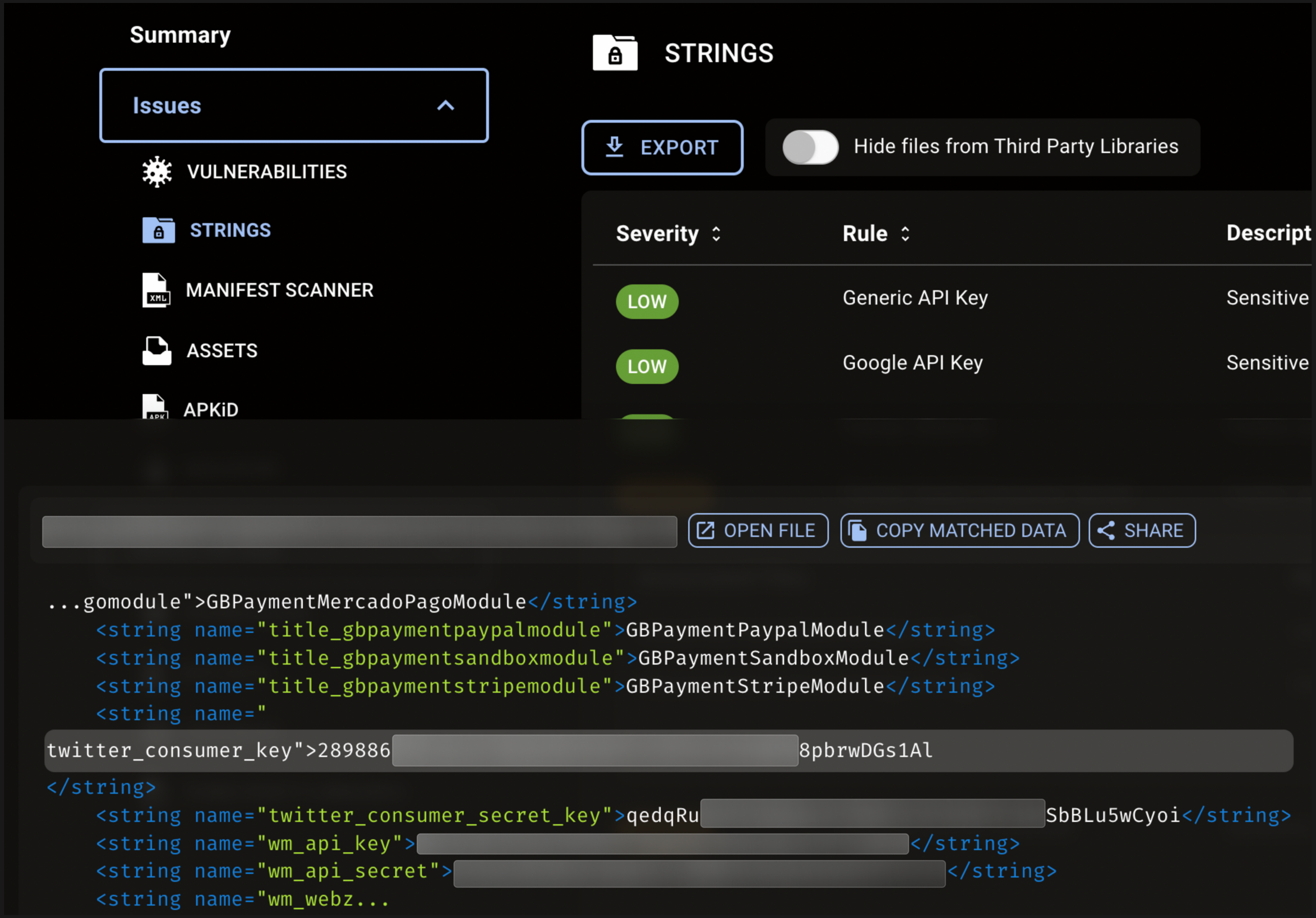
Alternate name used would be

Access token secret

Token Secret

resulting oauth_token_secret

Don't Complicate API Keys



Severity	Rule	Descript
LOW	Generic API Key	Sensitive
LOW	Google API Key	Sensitive

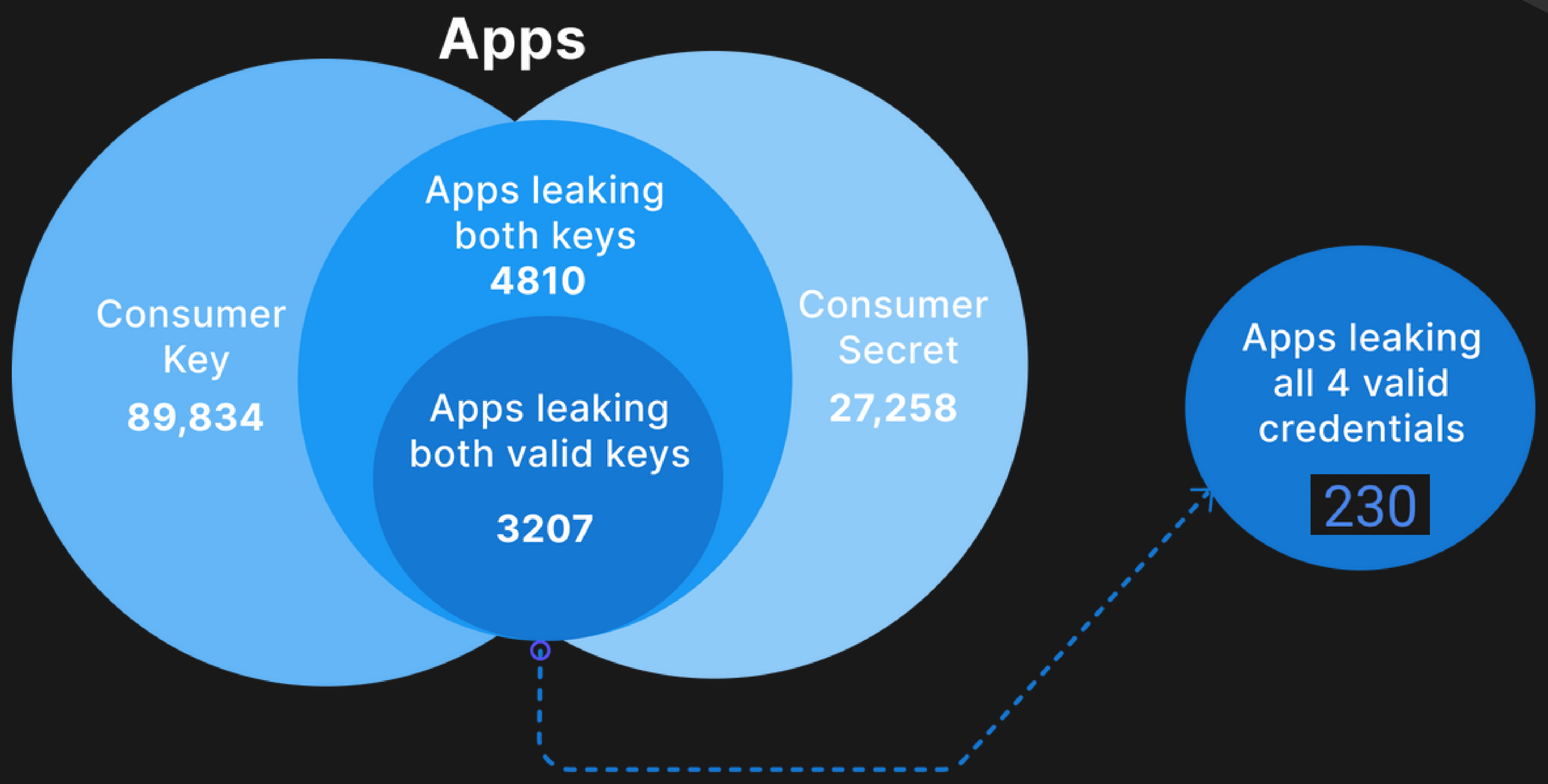
[redacted]

```
...gomodule">GBPaymentMercadoPagoModule</string>
    <string name="title_gbpaymentpaypalmodule">GBPaymentPaypalModule</string>
    <string name="title_gbpaymentsandboxmodule">GBPaymentSandboxModule</string>
    <string name="title_gbpaymentstripemodule">GBPaymentStripeModule</string>
    <string name="
twitter_consumer_key">289886[redacted]8pbrwDGs1Al
</string>
    <string name="twitter_consumer_secret_key">qedqRu[redacted]SbBLu5wCyoi</string>
    <string name="wm_api_key">[redacted]</string>
    <string name="wm_api_secret">[redacted]</string>
    <string name="wm_webz...
```

Impact of Four Keys Leaks

User-Based Authentication:

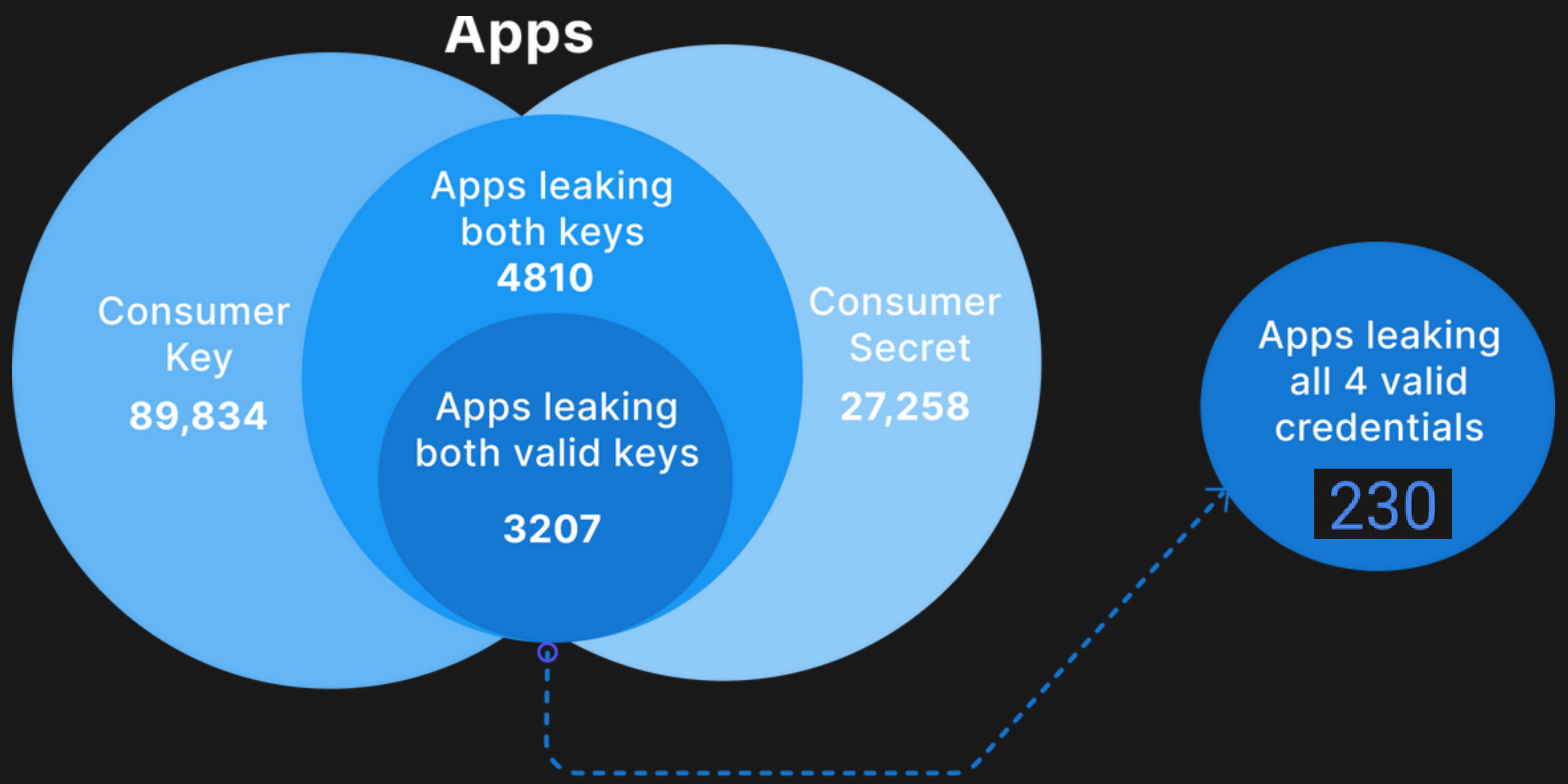
- Read DMs
- Retweet
- Like
- Delete
- Remove followers
- Follow any account
- Get account settings
- Change display picture



What about the Remaining Apps?

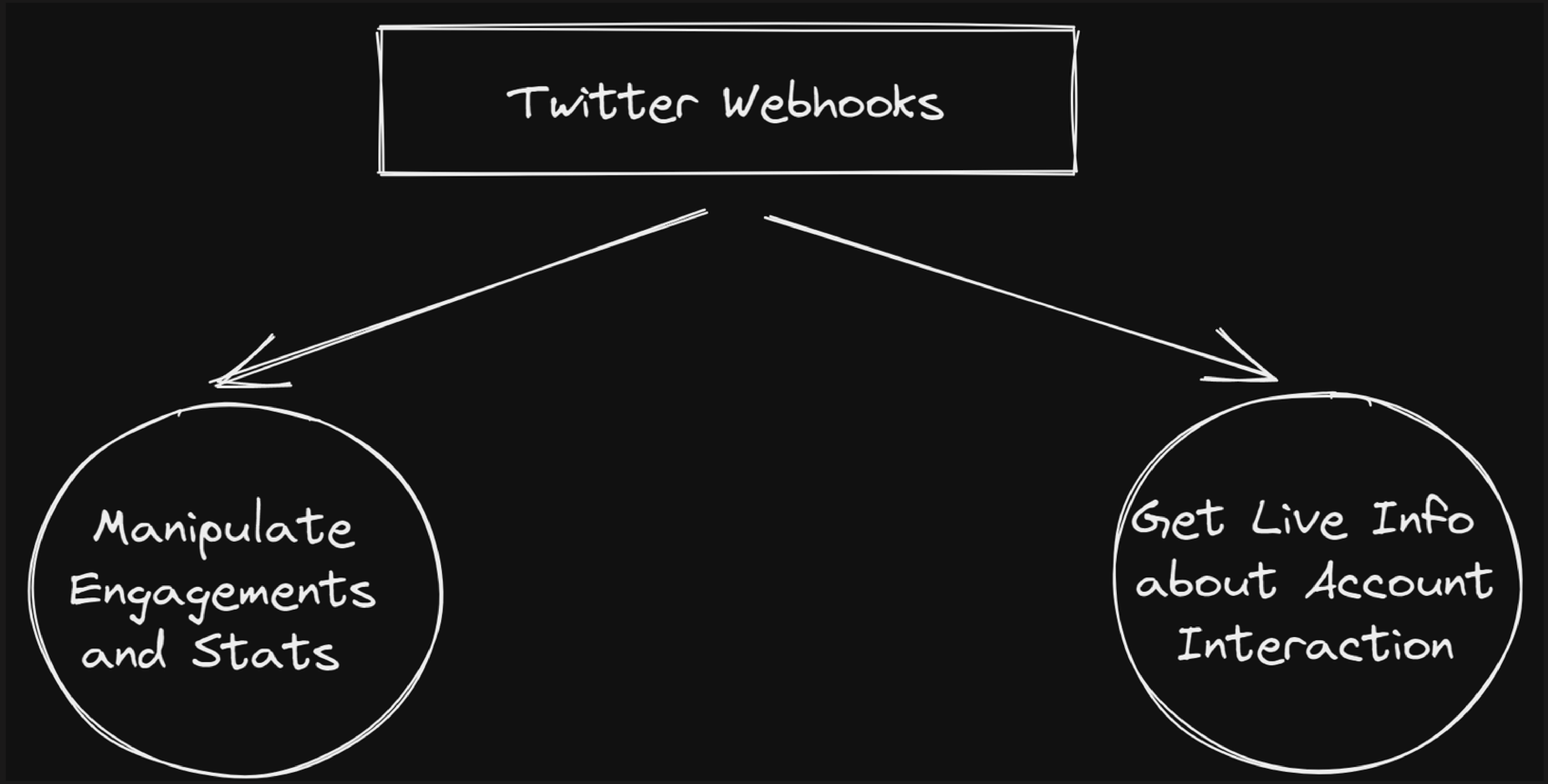
2977 apps were leaking only 2 keys [App-Based Auth]:

- Consumer Key
- Consumer Secret

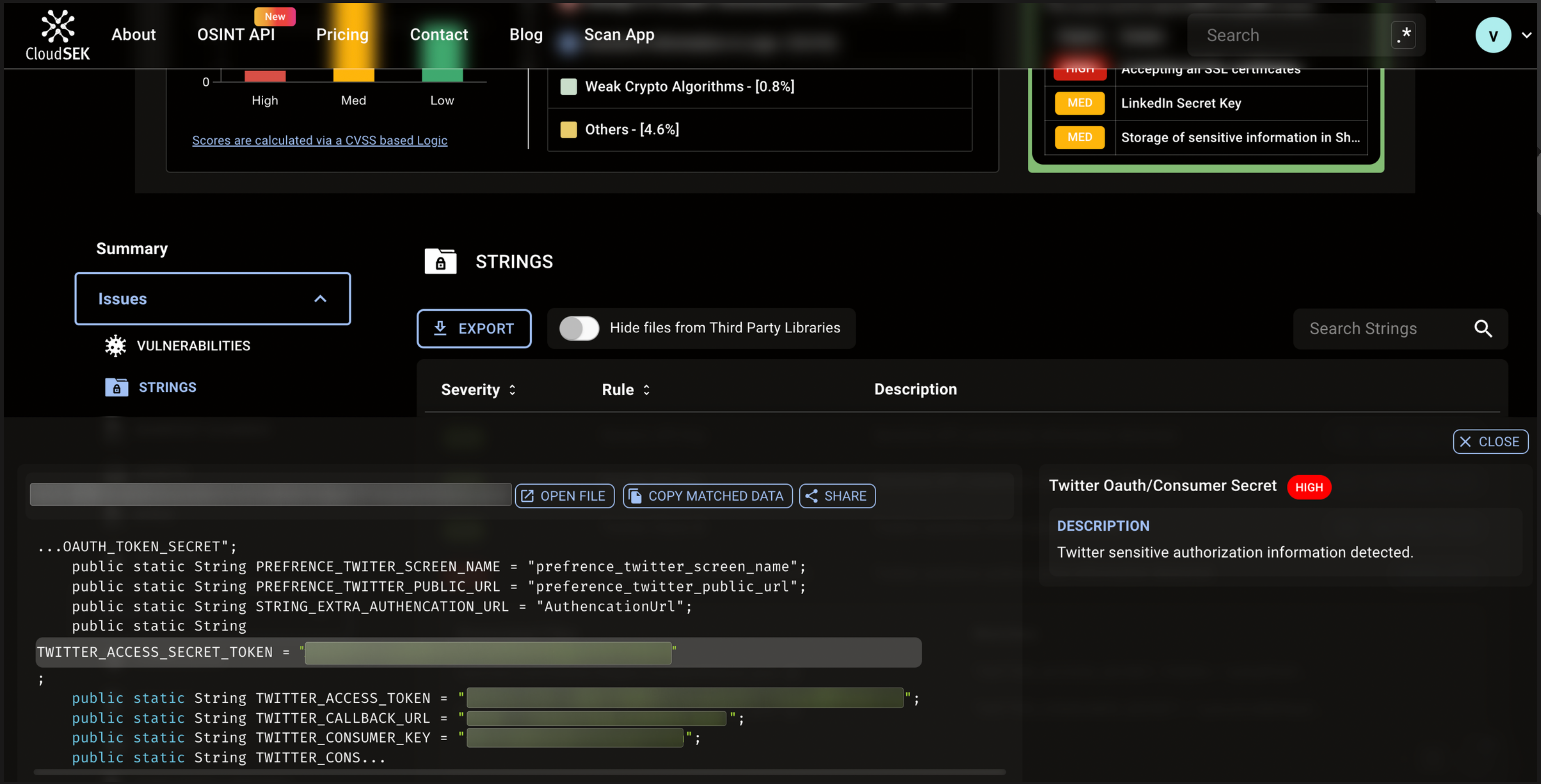

Apps
Consumer Key 89,834
Apps leaking both keys 4810
Consumer Secret 27,258
Apps leaking both valid keys 3207
Apps leaking all 4 valid credentials
230

What are Twitter Webhooks?

The use of Twitter webhooks requires OAuth 1.0a which is sometimes also referred to as “user context authentication" which allows to make API requests on behalf of a Twitter user. You will need Access to Premium/Enterprise Twitter API to use Webhooks.


Twitter Webhooks
Manipulate Engagements and Stats
Get Live Info about Account Interaction

EXPLOIT



Scores are calculated via a CVSS based Logic

- Weak Crypto Algorithms - [0.8%]
- Others - [4.6%]

HIGH	Accepting all SSL certificates
MED	LinkedIn Secret Key
MED	Storage of sensitive information in Sh...

STRINGS

Hide files from Third Party Libraries

Severity	Rule	Description

[redacted]

```
...OAUTH_TOKEN_SECRET";
    public static String PREFRENCE_TWITER_SCREEN_NAME = "prefrence_twitter_screen_name";
    public static String PREFRENCE_TWITTER_PUBLIC_URL = "preference_twitter_public_url";
    public static String STRING_EXTRA_AUTHENCATION_URL = "AuthencationUrl";
    public static String
TWITTER_ACCESS_SECRET_TOKEN = "[redacted]"
;
    public static String TWITTER_ACCESS_TOKEN = "[redacted]";
    public static String TWITTER_CALLBACK_URL = "[redacted]";
    public static String TWITTER_CONSUMER_KEY = "[redacted]";
    public static String TWITTER_CONS...
```

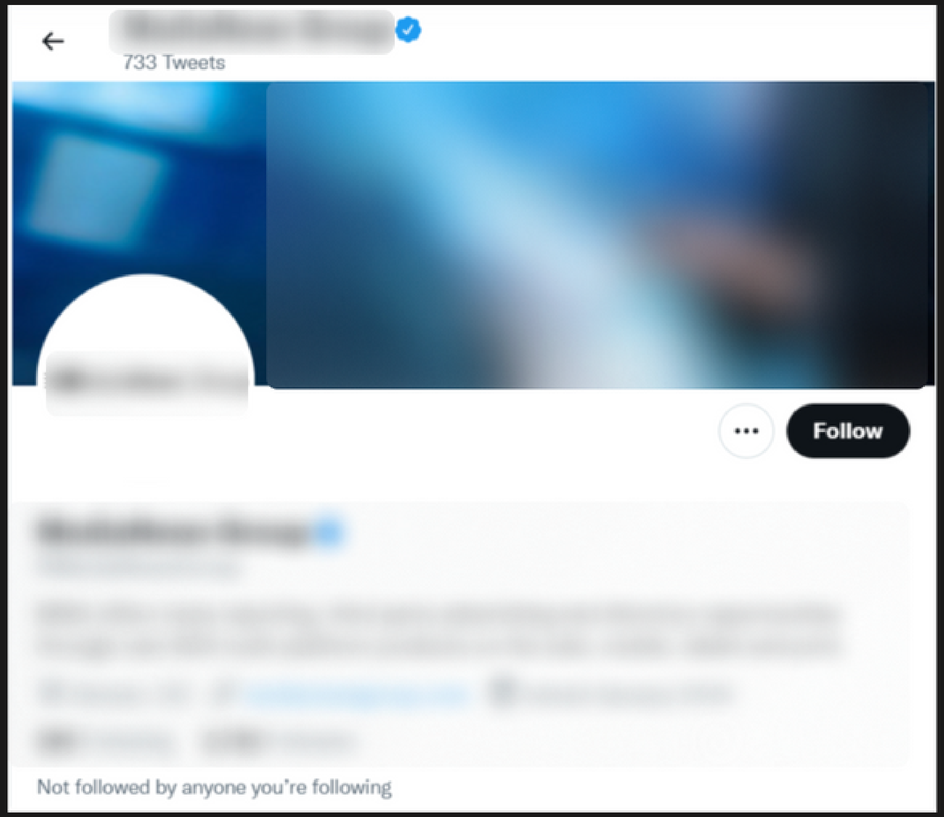
Twitter Oauth/Consumer Secret HIGH

DESCRIPTION

Twitter sensitive authorization information detected.

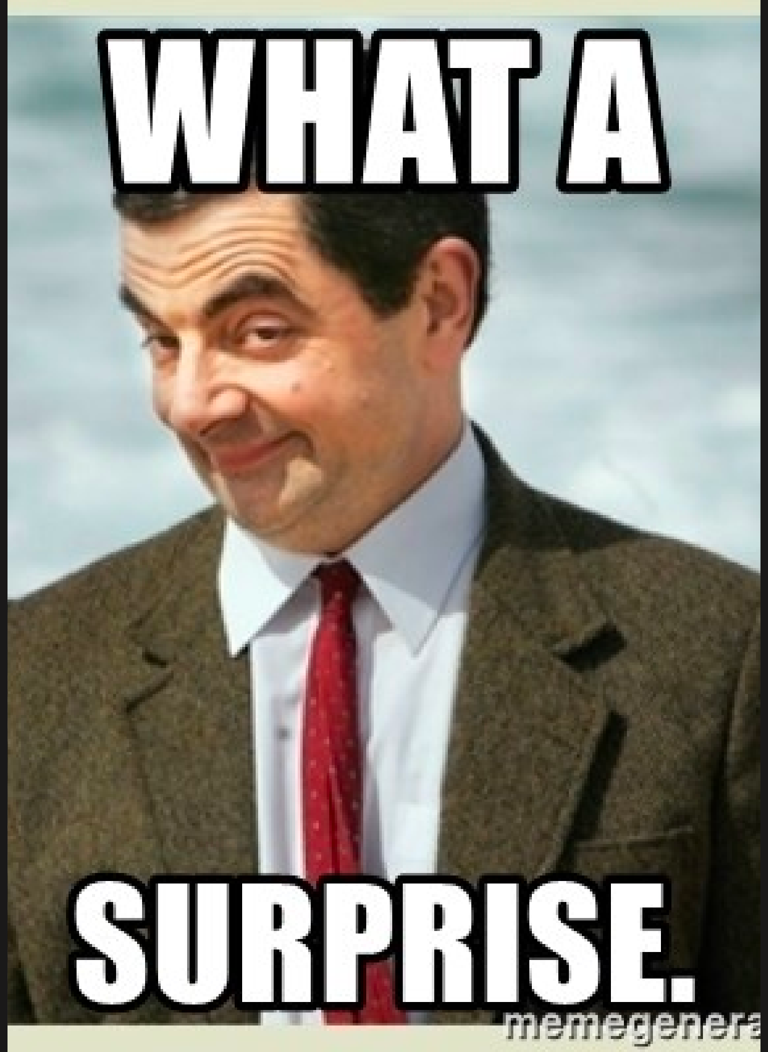
TA-DA Moment

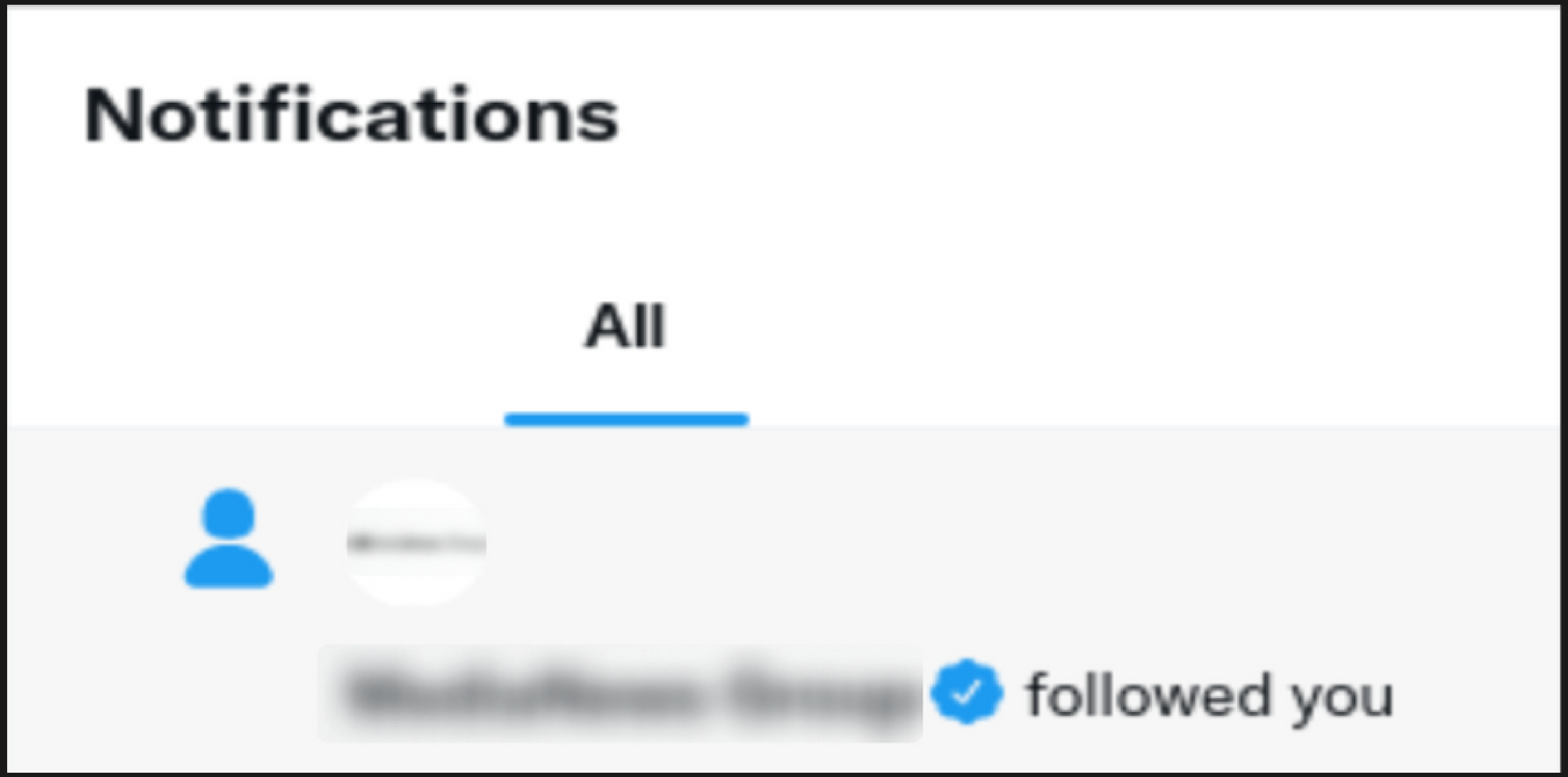
```
└─$ python3 twitter.py
```



733 Tweets

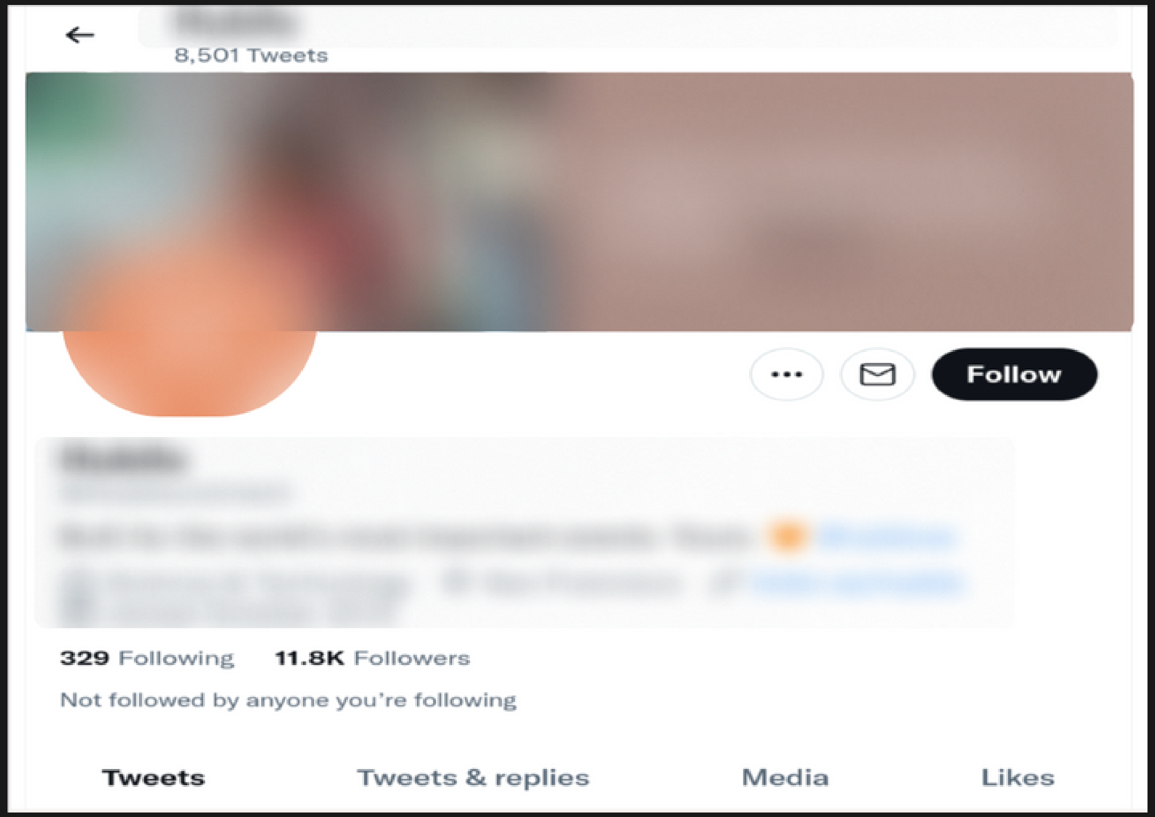
Not followed by anyone you're following





Notifications

followed you

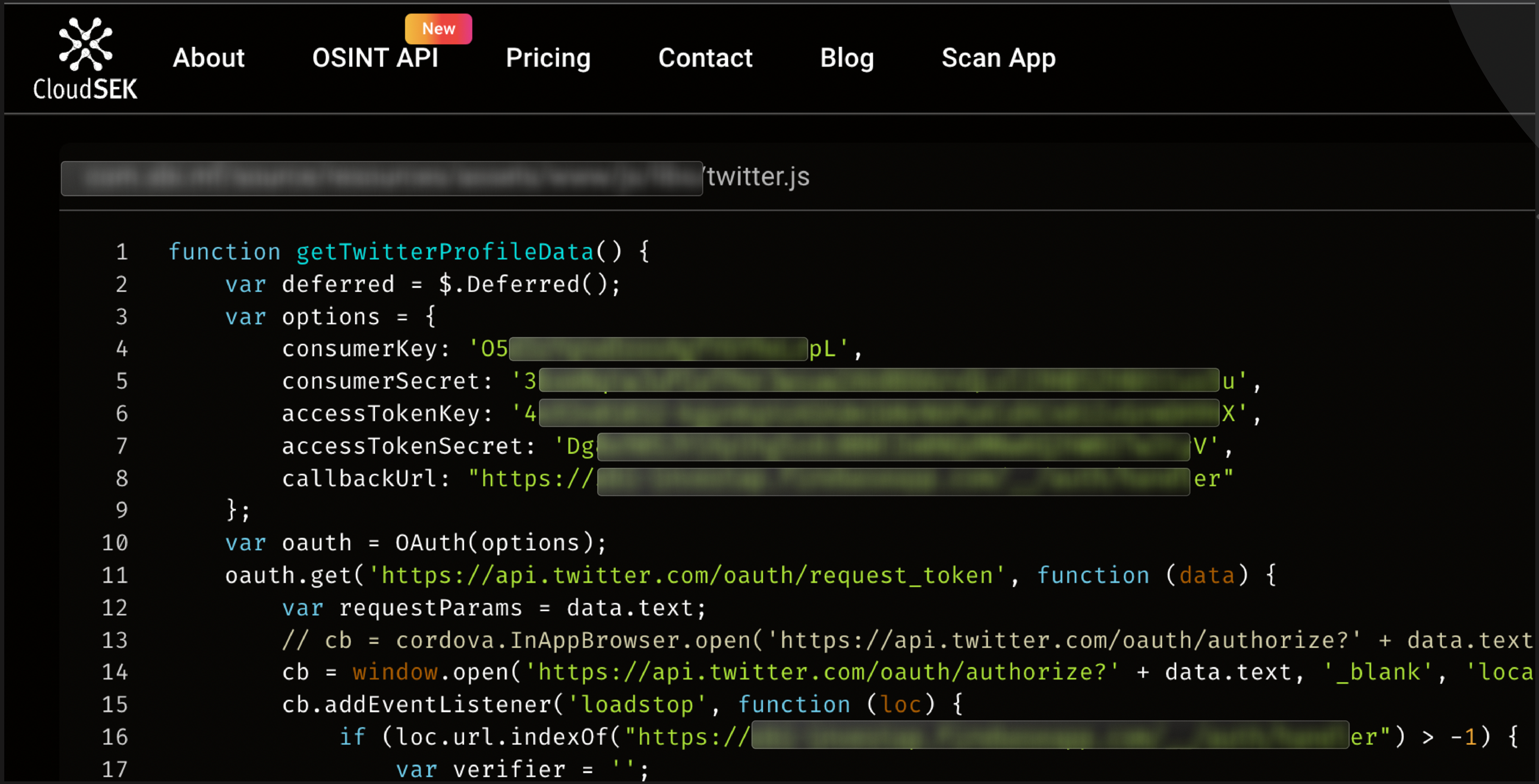


8,501 Tweets

329 Following 11.8K Followers

Not followed by anyone you're following

Not Limited To

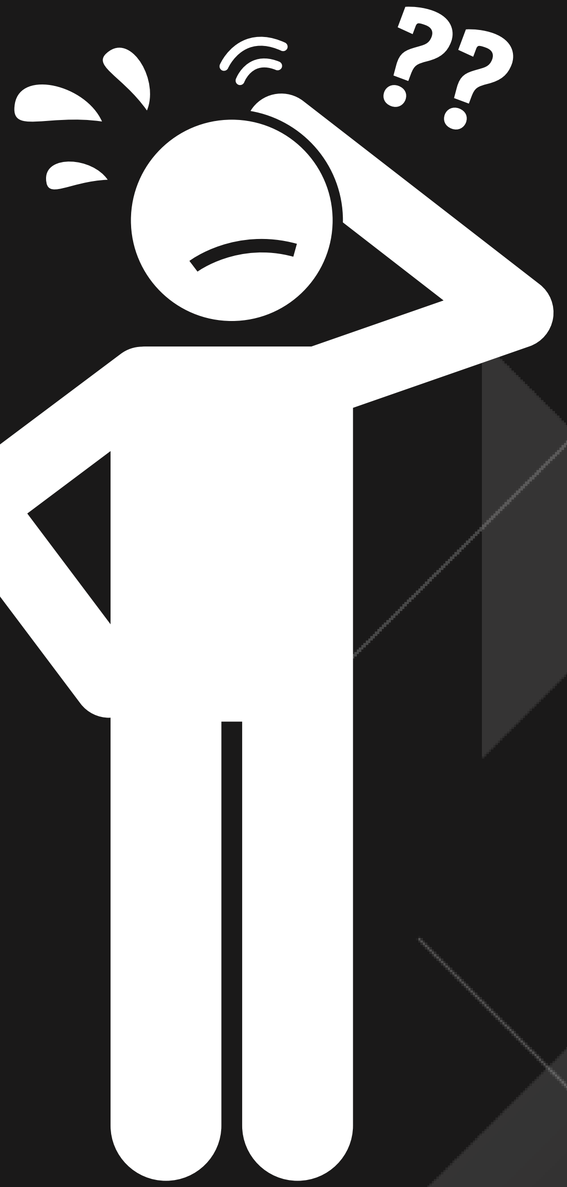


[redacted]/twitter.js

```
function getTwitterProfileData() {
    var deferred = $.Deferred();
    var options = {
        consumerKey: 'O5[redacted]pL',
        consumerSecret: '3[redacted]u',
        accessTokenKey: '4[redacted]X',
        accessTokenSecret: 'Dg[redacted]V',
        callbackUrl: "https://[redacted]er"
    };
    var oauth = OAuth(options);
    oauth.get('https://api.twitter.com/oauth/request_token', function (data) {
        var requestParams = data.text;
        // cb = cordova.InAppBrowser.open('https://api.twitter.com/oauth/authorize?' + data.text
        cb = window.open('https://api.twitter.com/oauth/authorize?' + data.text, '_blank', 'loca
        cb.addEventListener('loadstop', function (loc) {
            if (loc.url.indexOf("https://[redacted]er") > -1) {
                var verifier = '';
```

REMEDIATION

Where Problems Lies?



1

Security Pipeline

Pain of setting up a proper mobile app security testing pipeline while development.

2

Awareness

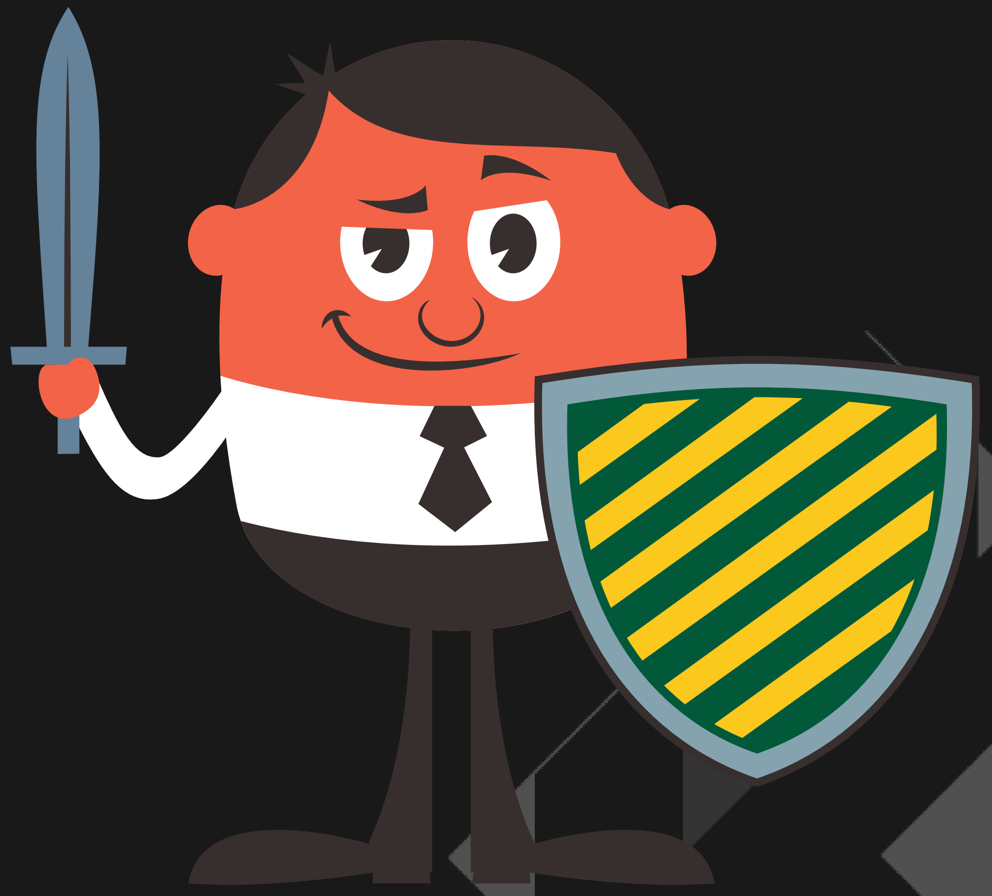
Lack of awareness on the scope/impact of the Hardcoded secret.

3

Budgeting

Companies not spending much on doing proper security testing on mobile apps - compared to web apps.

Defending against Attacks



Mitigation

Standardizing Review Procedures

Ensure accurate versioning. Publication requires the code base to be examined, reviewed, and approved prior to versioning. Complying with standardized procedures prevents key exposures.

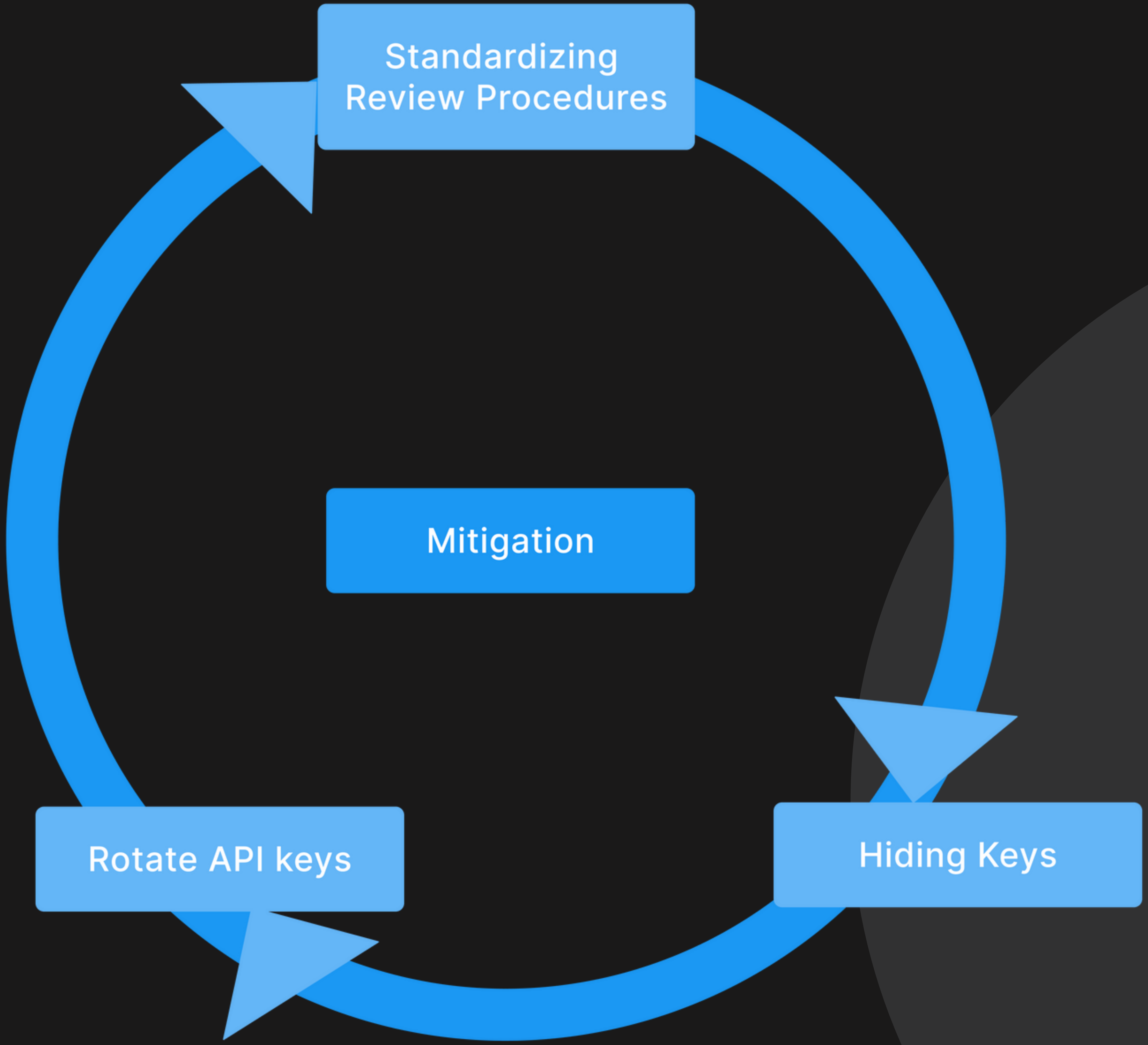
Hiding Keys

Variables in an environment are alternate means to refer to keys and disguise them.Variables save time and increase security. Adequate care should be taken to ensure that files containing environment variables in the source code are not included.

Rotate API keys

Rotating keys can help reduce the threat posed by leaked keys. Unused keys reduce the severity of invalidation. It is recommended to rotate keys every six months as existing keys get deactivated while new ones get generated.

Cyclic Process



CloudSEK

Thank you!

https://cloudsek.com/