



Targeted Attacks Supported by Recovered Data Structures of Web API's

Master Thesis - Introductory Presentation
Marc-Andrea Tarnutzer
23.10.2018

u^b



Networking Appified World





Problems

1. Software quality
 - Sloppy developers
 - Weak quality control
2. Software protection
 - Code availability
 - Endpoints / credentials exposure
3. App communications
 - Sensitive data
 - Flawed inputs



These Problems can lead to...

Data leaks / theft

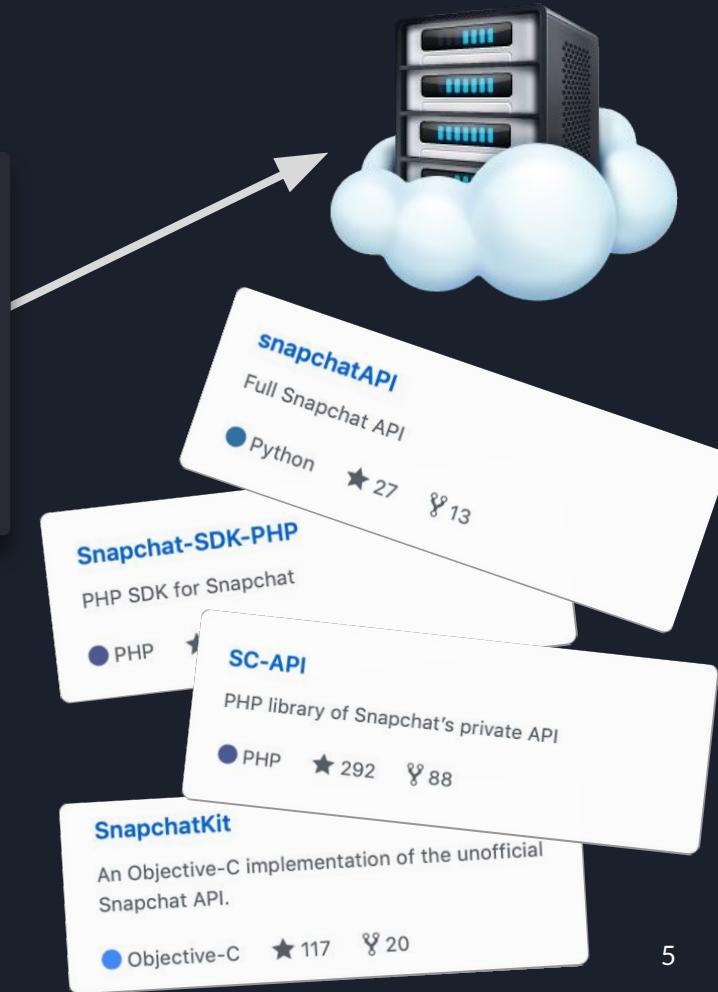
Denial of service

Impersonation / Unauthorized billing

Example: Snapchat



```
POST / HTTP/1.1
Host: api.snapchat.com
Authorization: Bearer eyJhbGciUxMiJ9.eyJzdWIiNTI1MTIyMj
Content-Type: application/json
{
    "to_id": "94857892",
    "message-type": "text",
    "created_at": "2018-09-18T13:32:39.708Z",
    "message": "Hey guys!"
}
```





Goals

Risk Assessment

Assess potential risks caused by client-server implementations

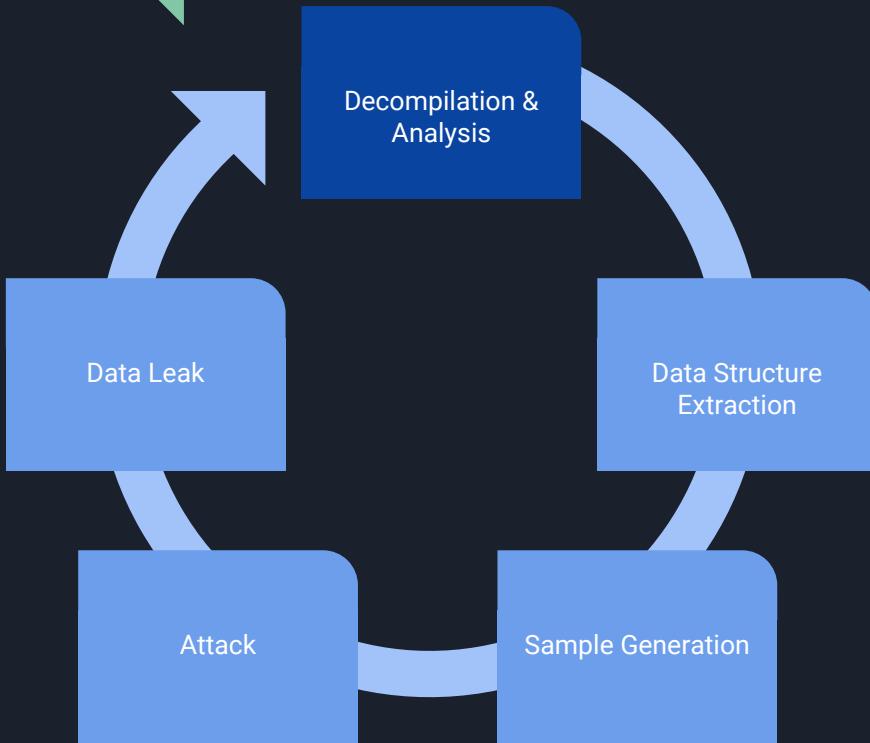
Automated App Analysis

Automated analysis of Android applications

Raise Awareness

Raise awareness of encountered threats

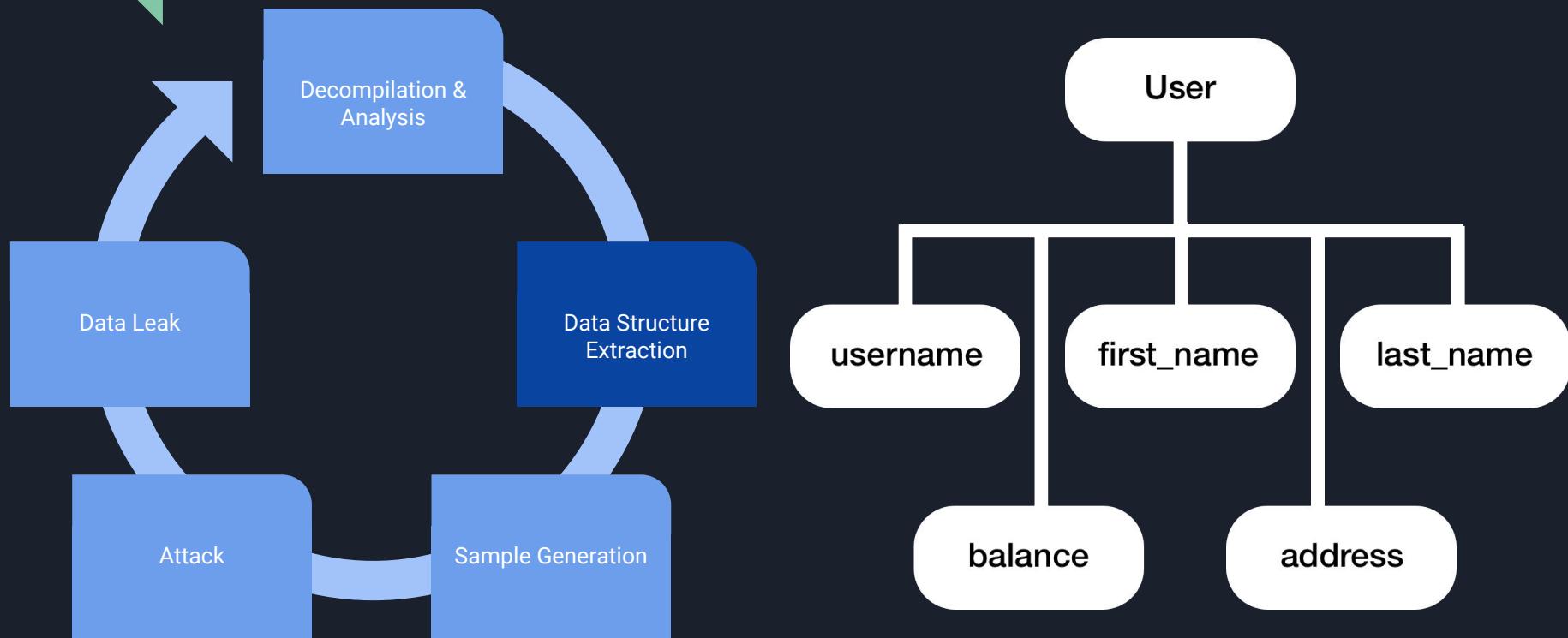
Automated App Analysis



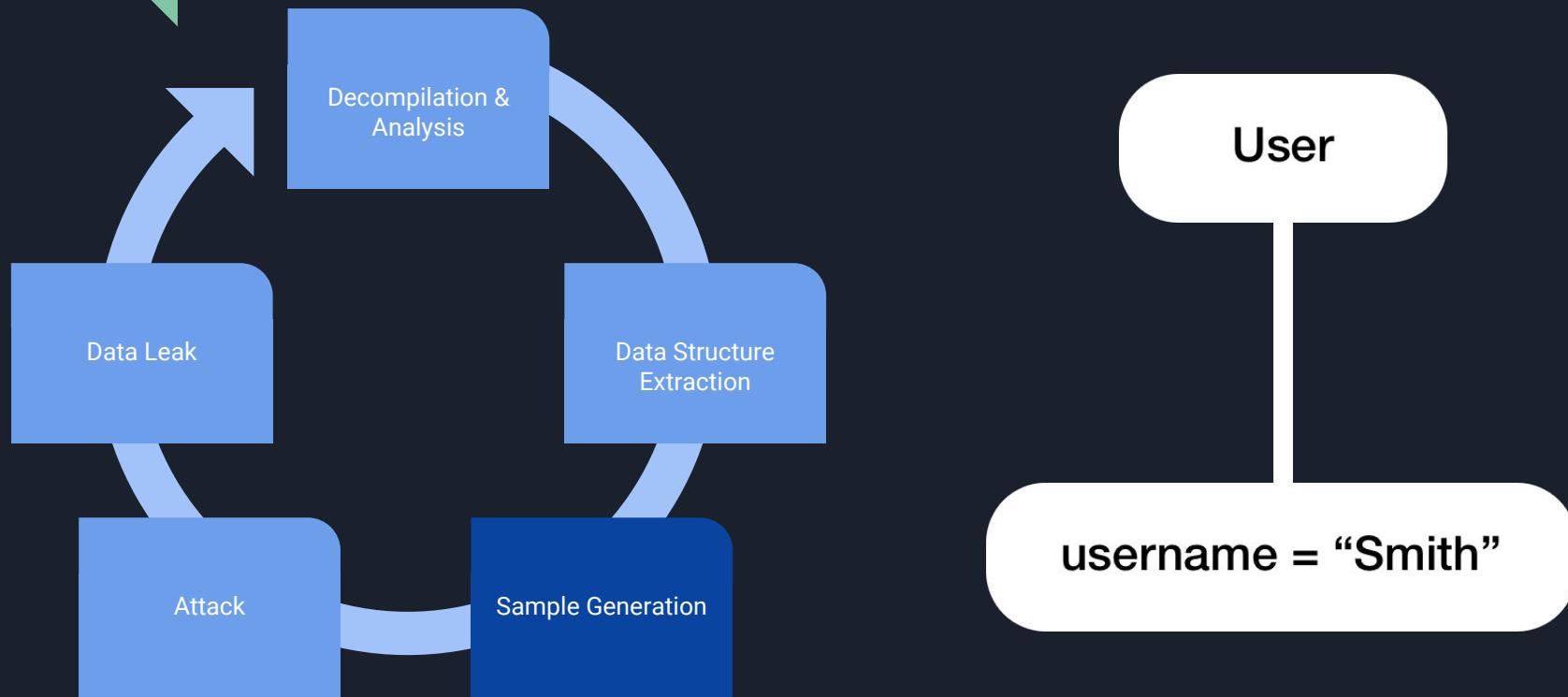
```
OkHttpClient client = new OkHttpClient();
Request request = new Request.Builder()
    .url(reqUrlString)
    .get().build();

client.newCall(request).enqueue(new Callback() {
    ...
    @Override
    public void onResponse(Call call,
        Response response) throws IOException {
        String response = response.body().string();
        JSONObject jObj = new JSONObject(response);
        User user = new User(
            jObj.getString("username"),
            jObj.getString("first_name"),
            jObj.getString("last_name"),
            jObj.getDouble("balance"),
            jObj.getString("address"));
    }
});
```

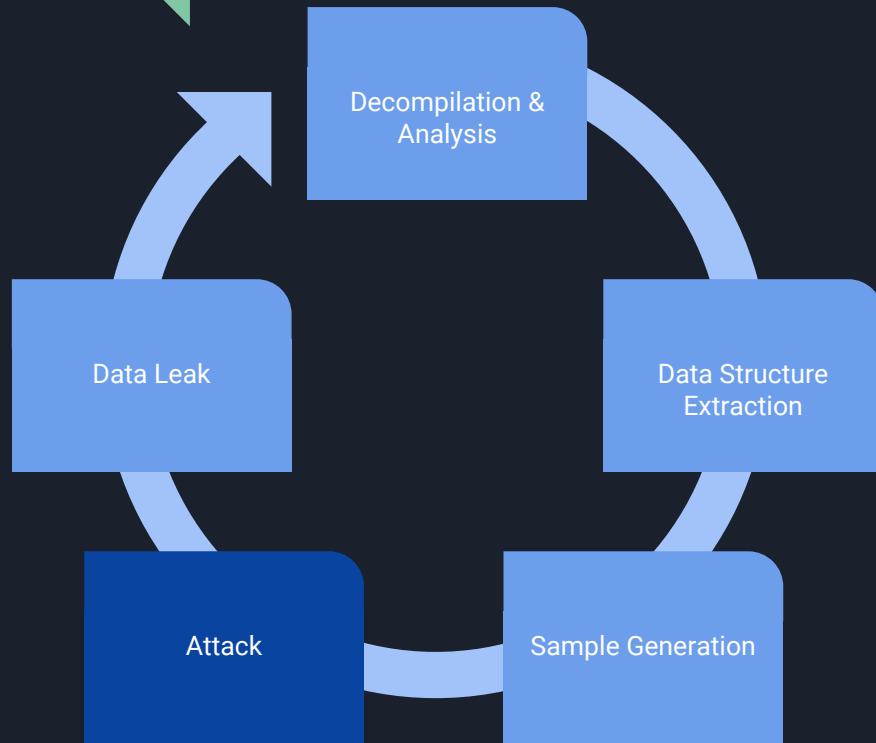
Automated App Analysis



Automated App Analysis



Automated App Analysis

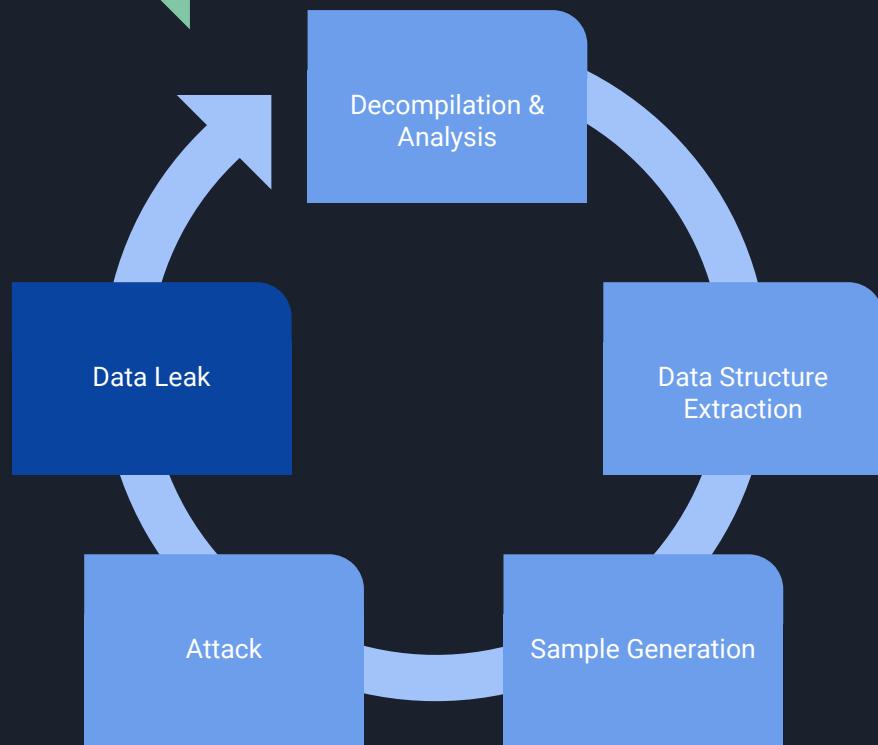


[https://bankname.com/
getDetails?username=Smith](https://bankname.com/getDetails?username=Smith)

- or -

predicted JSON object in header

Automated App Analysis



{

```
{"username": "Smith",  
 "first_name": "Mike",  
 "last_name": "Smith",  
 "balance": 9999999,  
 "address": "767 5th Ave  
 New York, NY 10153"}  
}
```

Progress

1 Implemented networking library search

2 Networking & data conversion libraries research

3 Implemented simple snippet search automation

4 AST traversal & symbol solving research

5 Implemented Android source code analyzer (Jandrolyzer)

Jandrolyzer: Library Support

The screenshot shows the Jandrolyzer interface with three main sections: a file tree on the left, a dependency graph in the center, and a list of dependencies on the right.

File Tree (Left):

- acr.browser.lightning_96_src.tar.gz
- app
 - src
 - LightningLite
 - LightningPlus
 - main
 - assets
 - java
 - res
 - AndroidManifest.xml
 - test
 - .gitignore
 - build.gradle
 - local.properties
 - proguard-project.txt
- gradle
 - .gitignore
 - .travis.yml
 - build.gradle
- CHANGELOG.md
- ic_launcher.png
- ic_launcher_small.png
- LICENSE
- local.properties
- README.md
- settings.gradle

```
dependencies {
    testCompile 'junit:junit:4.12'

    // support libraries
    def supportLibVersion = '25.4.0'
    compile "com.android.support:palette-v7:$supportLibVersion"
    compile "com.android.support:appcompat-v7:$supportLibVersion"
    compile "com.android.support:design:$supportLibVersion"
    compile "com.android.support:recyclerview-v7:$supportLibVersion"
    compile "com.android.support:support-v4:$supportLibVersion"

    // html parsing for reading mode
    compile 'org.jsoup:jsoup:1.10.2'

    // dependency injection
    def daggerVersion = '2.11'
    compile "com.google.dagger:dagger:$daggerVersion"
    annotationProcessor "com.google.dagger:dagger-compiler:$daggerVersion"
    provided 'javax.annotation:jsr250-api:1.0'

    // view binding
    def butterknifeVersion = '8.6.0'
    compile "com.jakewharton:butterknife:$butterknifeVersion"
    annotationProcessor "com.jakewharton:butterknife-compiler:$butterknifeVersion"

    // permissions
    compile 'com.anthonycr.grant:permissions:1.1.2'

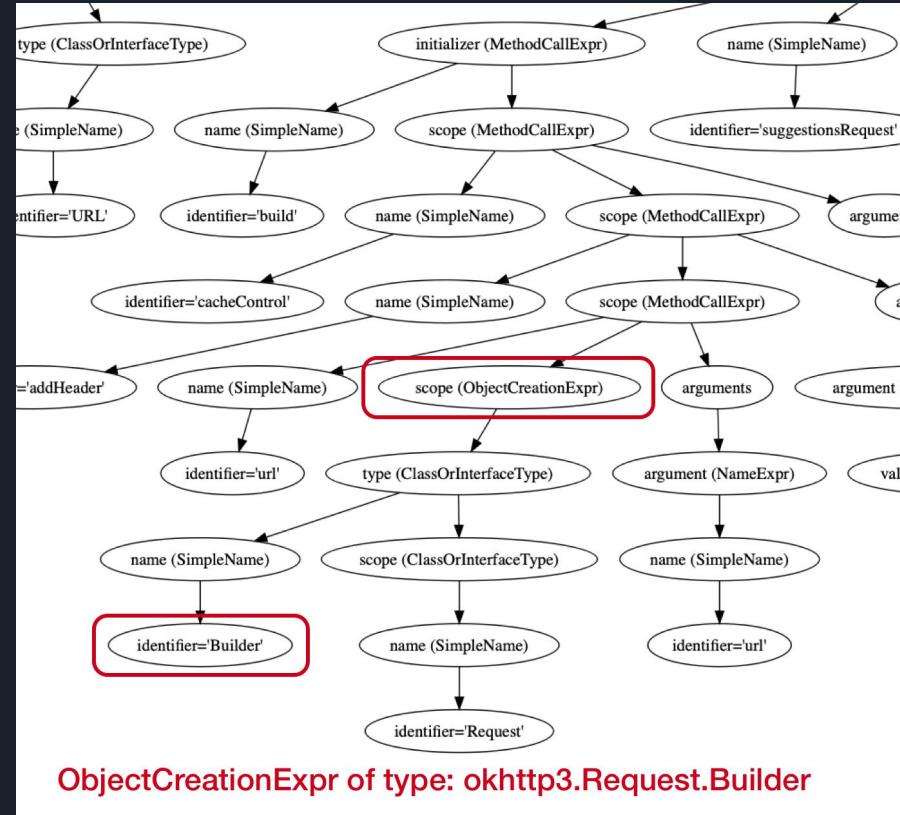
    // proxy support
    compile 'net.i2p.android:client:0.8'

    compile 'com.squareup.okhttp3:okhttp:3.8.0'
```

- com.squareup...khttp3-3.0.0
- com.squareup...khttp3-3.0.1
- com.squareup...khttp3-3.1.0
- com.squareup...khttp3-3.1.1
- com.squareup...khttp3-3.1.2
- com.squareup...khttp3-3.2.0
- com.squareup...khttp3-3.3.0
- com.squareup...khttp3-3.3.1
- com.squareup...khttp3-3.4.0
- com.squareup...khttp3-3.4.1
- com.squareup...khttp3-3.4.2
- com.squareup...khttp3-3.5.0
- com.squareup...khttp3-3.6.0
- com.squareup...khttp3-3.7.0
- com.squareup...khttp3-3.8.0
- com.squareup...khttp3-3.8.1
- com.squareup...khttp3-3.9.0
- com.squareup...khttp3-3.9.1
- com.squareup...http3-3.10.0
- com.squareup...http3-3.11.0
- com.squareup.retrofit-1.0.0
- com.squareup.retrofit-1.0.1
- com.squareup.retrofit-1.0.2
- com.squareup.retrofit-1.1.0
- com.squareup.retrofit-1.1.1
- com.squareup.retrofit-1.2.0

Jandrolyzer: AST Traversal

```
    ▼ reading
        ► activity
            /* ArticleTextExtractor.java
            /* Converter.java
            /* HtmlFetcher.java
            /* ImageResult.java
            /* JResult.java
            /* OutputFormatter.java
            /* SCache.java
            /* SHelper.java
    ▼ receiver
        /* NetworkReceiver.java
    ▼ search
        ► engine
        ▼ suggestions
            /* BaiduSuggestionsModel.java
            /* BaseSuggestionsModel.java *
            /* DuckSuggestionsModel.java
            /* GoogleSuggestionsModel.java
            /* SearchEngineProvider.java
            /* SuggestionsAdapter.java
            □ SuggestionsManager.kt
```



Jandrolyzer: Networking Snippet Example

```
@Nullable
private InputStream downloadSuggestionsForQuery(@NonNull String query, @NonNull String language) {
    String queryUrl = createQueryUrl(query, language);

    try {
        URL url = new URL(queryUrl);

        // OkHttp automatically gzips requests
        Request suggestionsRequest = new Request.Builder().url(url)
            .addHeader("Accept-Charset", mEncoding)
            .cacheControl(mCacheControl)
            .build();

        Response suggestionsResponse = mHttpClient.newCall(suggestionsRequest).execute();

        ResponseBody responseBody = suggestionsResponse.body();
        return responseBody != null ? responseBody.byteStream() : null;
    } catch (IOException exception) {
        Log.e(TAG, "Problem getting search suggestions", exception);
    }

    return null;
}
```



Challenges

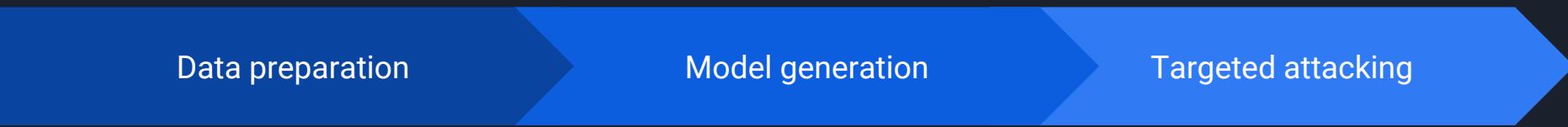
Memory

JavaParser expression resolving

Implementation variations



Next Steps



Data preparation

Model generation

Targeted attacking

Decompiled APK
analysis

Data structure
extraction

Endpoint extraction

Property evaluation
Sample generation

Automated endpoint
testing



Summary

Problems

Goals

Automated App Analysis

Progress

Jandrolyzer

Challenges

Next Steps