

Poseidon's Offspring: Charybdis and Scylla

 humansecurity.com/learn/blog/poseidons-offspring-charybdis-and-scylla



Researchers: Nico Agnese, Vikas Parthasarathy, Joao Santos, Adam Sell, Inna Vasilyeva

In this post:

- HUMAN's Satori Threat Intelligence & Research team discovered an operation we're calling **Scylla** (pronounced SILL-uh). It's the third wave of an attack we first reported in August 2019; the second wave, which we're calling **Charybdis** (pronounced kuh-RIB-diss), cropped up in late 2020.
- The attacks target a number of advertising SDKs within apps available via both Google's Play Store and Apple's App Store.
- Apps associated with the Scylla operation have been downloaded **13+ million** times.
- This attack is ongoing; HUMAN and the Satori team continue to pursue the operation and its perpetrators.

Introduction

"Modern defense" is the strategy behind protecting the internet and the advertising ecosystem from fraud through a combination of internet observability, collective protection, and actionable intelligence with disruptions of fraud operations when they're uncovered. This strategy is the basis of how HUMAN works to safeguard the internet from digital attacks, disrupt the economics of cybercrime, and reduce the cost of collective protection for customers.

Three years ago, in August of 2019, HUMAN's (then White Ops) [Satori team](#) reported about [a collection of 40+ Android apps openly committing multiple types of advertising fraud](#). That investigation, which we named Poseidon after the malicious code within the apps, resulted in Google removing the apps from their Play Store.

A disruption that results in dismantling the infrastructure powering a fraud scheme is rare. In the case of the original Poseidon operation, we weren't in a position to disrupt the infrastructure in that way, but we were able to work with the Google Play Store team to remove the malicious apps and ensure that the proliferation of that wave of the scheme stopped.

Poseidon is back again, unsurprisingly. This adaptation, which we've dubbed **Scylla**, showcases new tactics and techniques these threat actors have deployed to try and cover their tracks better than in earlier adaptations.

They have also expanded their attack to more parts of the digital advertising ecosystem, further underscoring the importance of collective protection to stop fraud at the source:

- The Scylla operation featured **75+** Android apps and **10+** iOS apps committing several flavors of ad fraud. These apps generated **13+ million** downloads in total before they were taken down.
- Satori partnered with the security teams at Google's Play Store and Apple's App Store. HUMAN provided indicators of compromise (IOCs) and other evidence involving the malicious Scylla code. Google and Apple responded promptly to the threat, expanding the ring of apps and removing them from their respective stores.
- HUMAN's customers are protected from attacks on the digital advertising ecosystem through our media security solutions, including MediaGuard.

Poseidon, the father

Before we can dive into the specifics of how the Charybdis and Scylla operations worked, it's helpful to have the context of how the original Poseidon phase of the scheme functioned and how the Satori team found it.

The 2019 Poseidon operation consisted of more than **40 Android apps openly committing ad fraud** within their code. The apps in question displayed ads out of context or hidden from view of the device user, both of which are classic mechanisms within the broader ad fraud category of Misleading User Interface (PDF).

What made the Poseidon operation interesting was its use of "receivers" to trigger these ads to load. Receivers are, generally speaking, useful little bits of code that tell an app when something has *changed* with respect to the device. For example, a receiver monitors when a phone has been put into Airplane Mode, and then instructs the app to stop trying to connect to the internet until Airplane Mode is turned off again.

However, well-intended and useful elements of code can be twisted to serve criminal goals. In the case of Poseidon, a receiver designed to determine if the phone was on and in use—a **USER.PRESENT** receiver—was key to instructing the host apps to call for ads to be shown out-of-context or hidden from the user's actual view.

The threat actors behind Poseidon didn't cover their tracks very well, though, as all of the code they built into the apps to handle the fraud was out in the open. Once the Satori team reverse-engineered the code segments named for Greek mythological figures—hence, "Poseidon"—the fraud tactics were entirely laid bare. There was no real attempt to obscure the intention of the code. That would change in future iterations of the scheme.

Poseidon, the father; Charybdis, the daughter

Following the original Poseidon attack, which we worked closely with Google's Play Store team to halt, the Satori team kept an eye on the threat actors to see how they would respond.

This phase of fraud-fighting, the adaptation phase, is a crucial part of changing the economics of cybercrime. Every fraud scheme is a race against time for a threat actor: the fraudsters only have until their scheme is shut down to reap enough profit to recoup their cost of developing and deploying it in the first place. The more that organizations can work together and share information, the more threat-hunters like Satori can discover these fraud schemes and shrink the criminal's profit window. Eventually, the cycle of operation discovery and dismantling becomes so short that fraudsters won't have enough time to recoup even their costs, so they'll move on to another target.

In the later part of 2020 and early into 2021, the Satori team uncovered the second wave of this threat actor's scheme. This iteration, which we've dubbed Charybdis (named for Poseidon's daughter), evolved from the Poseidon operation in two key areas:

1. **Code Obfuscation:** The threat actors leveraged a popular developer tool named **Allatori**, which swaps out elements of code for single letters or numbers, making it very difficult for someone to look at the raw code of an app to understand what exactly it's supposed to do. There are many legitimate uses for tools like Allatori—much like there are legitimate ways to use many receivers—for example, the developer of a popular app might want to slow down the development of knockoff apps that eat into download numbers and sales. But—much like the case with the use of receivers—the intent is what's most important. In the case of Charybdis and Scylla, the fraudsters use Allatori to *hide the code committing the fraud inside the host apps*.
2. **SDK Targeting:** The original Poseidon scheme did not target any specific advertising platform via the code buried inside the host apps. With Charybdis, however, Satori researchers were able to reverse the obfuscated code and discovered targeting of specific well known advertising platforms and systems.

Poseidon, the father; Charybdis, the daughter; Scylla, the granddaughter

HUMAN's Satori team has uncovered a collection of **75+ apps on Google's Play Store** and **10+ apps on Apple's App Store** that contained obfuscated code similar to Charybdis. This third and latest phase of the operation is named for Scylla, Poseidon's granddaughter and a sea monster like Charybdis. These apps, when installed, commit several different flavors of advertising fraud:

- **App and Bundle ID Spoofing:** this is when an app contains code that tells advertisers and ad tech companies that it's a different app entirely, with the intent of tricking those companies into placing ads there (if it's pretending to be an especially popular app) or into spending more money to place ads (if it's pretending to be something else entirely, like a streaming service).

The Scylla apps contained code that pretended to be other, legitimate games for advertising purposes, helping to keep their operation quiet.

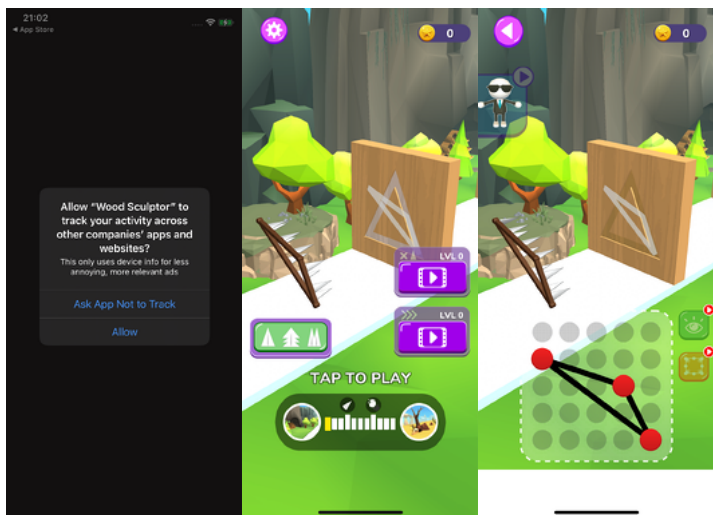
- **Out of Context (OOC) and Hidden Ads:** this is when ads are shown in a situation where ads should not be shown, or are rendered in a way that a user cannot possibly see them. For example, if your phone randomly displays an ad on your home screen, that's an out of context ad that a bad app is prompting. Alternatively, sometimes, apps will try and sneak extra ads in but not *show* them to a user. The ads still count as having been "viewed" with the fraudulent code faking viewability metrics, which means the fraudster makes money, but no ad was actually visible.

In the case of Scylla, the Receiver code (described above) would trigger when the apps weren't open. For example, triggering when a device was simply unlocked on the home screen.

- **Fake Clicks:** Every advertiser knows that clicks are more valuable than impressions. And many advertising programs are set up to bill that difference accordingly. So, naturally, fraudsters are eager to develop ways to make it look like the ads being shown are also being clicked; they make more money as a result.

Code within the Scylla apps is built to take information from *real* clicks—like the timing of a real person's click of an ad and where on the ad they clicked it—and then send that information again as a *fake* click to cash in.

Also new in the Scylla attack is the expansion from targeting advertising SDKs focused on Android apps into targeting iOS apps as well:



Wood Sculptor, an iOS app associated with Scylla

Source: Satori Threat Intelligence and Research Team

Above, you can see a game called Wood Sculptor. The app is a knock-off of a popular game in which the user slices a path on a flat surface to allow a predetermined shape to pass through the resulting opening. **Satori researchers who downloaded Wood Sculptor never saw any ads displayed during the game's operation.**

488	https://	POST	/operative/RV_1	✓	200	147		
489	https://	POST	/v6/games/3956050/requests?idfi=d2...	✓	200	22025	JSON	
490	https://	GET	/assets/6183daa898127d145cbfc28c/...		200	2575849	video	mp4
491	https://	GET	/load?account_id=27a73701115b471c...	✓	200	359	JSON	
492	https://	POST	/operative/RV_1	✓	200	147		
493	https://	POST	/v6/games/3956050/requests?idfi=d2...	✓	200	367	JSON	
494	https://	GET	/load?account_id=27a73701115b471c...	✓	200	359	JSON	
495	https://	GET	/store-icons/6d83418e-320a-4bec-8b...	✓	200	22600	JPEG	jpg
496	https://	POST	/v1/category/experiment	✓	200	1239	JSON	
497	https://	GET	/load?account_id=27a73701115b471c...	✓	200	359	JSON	
498	https://	POST	/v6/games/3956050/requests?idfi=d2...	✓	200	367	JSON	
499	https://	GET	/assets/5fc8ce0bdfc31fe7901b6a9a/7...	✓	200	2837773	HTML	html
500	https://	GET	/load?account_id=27a73701115b471c...	✓	200	359	JSON	
501	https://	POST	/m/ad	✓	200	17705	JSON	
502	https://	POST	/v6/games/3956050/requests?idfi=d2...	✓	200	21351	JSON	
503	https://	GET	/4.0/ad?sdk_version=6.15.2&tz_offset...	✓	200	604	JSON	0/ad
504	https://	GET	/load?account_id=27a73701115b471c...	✓	200	359	JSON	
505	https://	GET	/4.0/ad?sdk_version=6.15.2&tz_offset...	✓	200	604	JSON	0/ad
506	https://	GET	/load?account_id=27a73701115b471c...	✓	200	359	JSON	
507	https://	GET	/4.0/ad?sdk_version=6.15.2&tz_offset...	✓	200	604	JSON	0/ad
508	https://	GFT	/load?account_id=27a73701115b471c...	✓	200	359	JSON	
509	https://	GET	/load?account_id=27a73701115b471c...	✓	200	359	JSON	
510	https://	POST	/m/imp	✓	200	335	text	
511	https://	POST	/api/v5/ads	✓	200	13292	JSON	
512	https://	GET	/template-localization/v18n.min.js	✓	200	5139	script	js

Packet capture from Wood Sculptor

Source: Satori Threat Intelligence and Research Team

Above, a packet capture of activity from Wood Sculptor. Note the numerous calls to ad servers. Below, execution traces show additional web views where the ads called above are rendered invisible to the user.

```

UIWindow: 0x15d7f1870; frame = (0 0; 375 812); gestureRecognizers = <NSArray: 0x282a3c420>; layer = <UIWindowLayer: 0x282411e0>
  <UIWebView: 0x1657a90a0; frame = (0 0; 1024 768); clipsToBounds = YES; hidden = YES; layer = <CALayer: 0x28251b220>
    <WKWebView: 0x154ee200; frame = (512 384; 0 0); layer = <CALayer: 0x2825191a0>
      <WKScrollWebView: 0x1520b0400; baseClass = UIScrollView; frame = (0 0; 0 0); clipsToBounds = YES; gestureRecognizers = <NSArray: 0x2828201e0>; layer = <CALayer: 0x28251a900>; contentOffset: (0, 0); contentSize: (0, 0); adjustedContentInset: (0, 0, 0, 0)
        <WKContentView: 0x155470600; frame = (0 0; 0 0); anchorPoint = (0, 0); gestureRecognizers = <NSArray: 0x282821860>; layer = <CALayer: 0x28251e0e0>
          <UIWebView: 0x1657cc00; frame = (0 0; 0 0); anchorPoint = (0, 0); clipsToBounds = YES; layer = <CALayer: 0x282510e00>
            <UIWebView: 0x1657a8850; frame = (0 0; 0 0); autoresize = WH; layer = <CALayer: 0x282510e00>
              <WKCompositingView: 0x1657058c0; frame = (0 0; 0 0); layer = <CALayer: 0x282506000>; layerID = 1 "drawing area root"
                <WKCompositingView: 0x16570550; frame = (0 0; 0 0); anchorPoint = (0, 0); clipsToBounds = YES; layer = <CALayer: 0x282506000>; layerID = 2 "content root"
                  <WKCompositingView: 0x165705750; frame = (0 0; 0 0); layer = <CALayer: 0x282506000>; layerID = 5 "RenderView 0x180ba000"
                    <WKCompositingView: 0x1657d2560; frame = (0 0; 0 0); layer = <CALayer: 0x282506000>; layerID = 6 "TileGrid container"
                      <WKCompositingView: 0x1657b5b0; frame = (0 0; 0 0); layer = <CALayer: 0x282506000>; layerID = 7 "Page TiledBacking containment"
                        <WKCompositingView: 0x15d12850; frame = (0 0; 0 0); layer = <CALayer: 0x282506000>; layerID = 3 "Document overlay container"
                          <UILayerHostView: 0x16570b50; frame = (0 0; 0 0); layer = <CALayerHost: 0x282506000>
                            <UIWebView: 0x16570a00; frame = (0 0; 0 0); anchorPoint = (0, 0); opaque = NO; layer = <CALayer: 0x28251b440>
                              <UITextSelectionView: 0x16570a00; frame = (0 0; 0 0); userInteractionEnabled = NO; layer = <CALayer: 0x2825146c0>
                                <UIWebView: 0x16570a00; frame = (0 0; 0 0); userInteractionEnabled = NO; layer = <CALayer: 0x2825143c0>
                                  <UIScrollViewScrollIndicator: 0x1657ad20; frame = (-39 -6; 36 3); alpha = 0; autoresize = TM; layer = <CALayer: 0x282504700>
                                    <UIWebView: 0x16570a00; frame = (0 0; 36 3); layer = <CALayer: 0x282504700>
                                      <UIScrollViewScrollIndicator: 0x16570e20; frame = (-6 -39; 3 36); alpha = 0; autoresize = LM; layer = <CALayer: 0x282504700>
                                        <UIWebView: 0x16570a00; frame = (0 0; 3 36); layer = <CALayer: 0x282504700>
                                          <WKWebView: 0x15212a00; frame = (512 384; 0 0); layer = <CALayer: 0x282504880>
                                            <WKScrollWebView: 0x15042f600; baseClass = UIScrollView; frame = (0 0; 0 0); clipsToBounds = YES; gestureRecognizers = <NSArray: 0x282827e40>; layer = <CALayer: 0x2825040c0>; contentOffset: (0, 0); contentSize: (0, 0); adjustedContentInset: (0, 0, 0, 0)
                                              <WKContentView: 0x15e008000; frame = (0 0; 0 0); anchorPoint = (0, 0); gestureRecognizers = <NSArray: 0x282823ba0>; layer = <CALayer: 0x282504a00>
                                                <UIWebView: 0x1657af400; frame = (0 0; 0 0); anchorPoint = (0, 0); clipsToBounds = YES; layer = <CALayer: 0x2825040c0>
                                                  <UIWebView: 0x1657af30; frame = (0 0; 0 0); autoresize = WH; layer = <CALayer: 0x2825040c0>
                                                    <WKCompositingView: 0x16517c710; frame = (0 0; 0 0); layer = <CALayer: 0x2825095e0>; layerID = 1 "drawing area root"
                                                      <WKCompositingView: 0x16516930; frame = (0 0; 0 0); anchorPoint = (0, 0); layer = <CALayer: 0x2825090c0>; layerID = 8 "content root"
                                                        <WKCompositingView: 0x16517f3b0; frame = (0 0; 0 0); layer = <CALayer: 0x2825090c0>; layerID = 9 "RenderView 0x18667900"
                                                          <WKCompositingView: 0x16516c00; frame = (0 0; 0 0); layer = <CALayer: 0x2825090c0>; layerID = 10 "TileGrid container"
                                                            <WKCompositingView: 0x1651ab00; frame = (0 0; 0 0); layer = <CALayer: 0x2825090c0>; layerID = 11 "Page TiledBacking containment"
                                                              <UILayerHostView: 0x16570a00; frame = (0 0; 0 0); layer = <CALayerHost: 0x2825061e0>
                                                                <UIWebView: 0x16570a00; frame = (0 0; 0 0); anchorPoint = (0, 0); opaque = NO; layer = <CALayer: 0x282509f00>
                                                                  <UIScrollViewScrollIndicator: 0x1651ac00; frame = (-39 -6; 36 3); alpha = 0; autoresize = TM; layer = <CALayer: 0x282508060>
                                                                    <UIWebView: 0x16570a00; frame = (0 0; 36 3); layer = <CALayer: 0x282508060>
                                                                      <UIScrollViewScrollIndicator: 0x1651ac30; frame = (-6 -39; 3 36); alpha = 0; autoresize = LM; layer = <CALayer: 0x282509c00>
                                                                        <UIWebView: 0x16570a00; frame = (0 0; 3 36); layer = <CALayer: 0x282508200>
                                                                          <UITransitionView: 0x15d7f020; frame = (0 0; 375 812); autoresize = WH; layer = <CALayer: 0x282567180>
                                                                            <UIBackdropView: 0x1630e0910; frame = (0 0; 375 812); autoresize = WH; layer = <CALayer: 0x282567180>
                                                                              <UIWebView: 0x15d7f1e70; frame = (0 0; 375 812); autoresize = WH; layer = <CALayer: 0x282412480>
                                                                                <UIWebView: 0x1652f210; frame = (0 0; 0 0); hidden = YES; userInteractionEnabled = NO; layer = <CALayer: 0x2825e9980>

```

UI elements identifying the location of webviews for ads rendered through Wood Sculptor

Source: Satori Threat Intelligence and Research Team

How Scylla Worked

Spoofting

Many of the operations that our Satori team has uncovered use [app or bundle ID spoofing](#) as a primary fraud mechanism. Our [PARETO](#) investigation, for example, uncovered 29 Android apps that were pretending to be more than 6,000 CTV-based apps, which generally carry higher prices for advertisers than the average mobile game.

In layman's terms, the threat actors code their apps to pretend to be *other* apps for advertising purposes, often because the app they're pretending to be is worth more to an advertiser than the app would be by itself. They do this by inserting a different "bundle ID" or "app ID" (think driver's license) in the code. Additionally, since some ad verification platforms can catch tactics like these by looking for frequently-faked bundle IDs, the threat actors will cycle through these IDs to avoid getting caught.

The apps in the Scylla operation are instructed which bundle ID to use by a remote [command-and-control](#) (C2) server, which tells the app which bundle ID to dynamically insert in the code for that specific impression opportunity, all in a matter of milliseconds.

```



1 POST /mediation?adUnit=3 HTTP/1.1
2 Content-Type: application/json
3 User-Agent: Dalvik/2.1.0 (Linux; U; Android 10; Redmi 5 Plus Build/QQ1B.200105.004)
4 Content-Length: 1251
5 Host:
6 Connection: close
7 Accept-Encoding: gzip, deflate
8
9 {
  "isLimitAdTrackingEnabled":false,
  "appVersion":"1.0.2",
  "tz":"Europe/Lisbon",
  "icc":"pt",
  "advertisingId":"2138fbeb-7d5a-42c5-a395-a69a8d605597",
  "language":"en",
  "battery":100,
  "mcc":0,
  "connectionType":"wifi",
  "internalFreeMemory":51003,
  "osVersion":"29(10)",
  "appKey":"133dbale9",
  "firstSession":"true",
  "deviceOEM":
  "build":
  "mnc":0,
  "deviceOS":"Android",
  "bundleId":"com.painting.war.inpaper",
  "sessionId":
  "externalFreeMemory":51003,
  "advertisingIdType":"UUID",
  "jb":"false",
  "sdkVersion":"7.1.5.1",
  "deviceModel":
  "getMinutesOffset":0,
  "userIdType":"userGenerated",
  "userId":
  "timestamp":
  "adUnit":3,
  "events":[]
}

```

Bundleid spoofing within Scylla

Source: Satori Threat Intelligence and Research Team

The above screenshot is taken from a game called "Lady Run". The bundle ID, however, names a different app entirely, a game called "War in Painting".

	22	http://	/b/c?a=jshV4Lc%2BYGc7B9ImKf0gSX1rZH9...	HTTP/1.1	200
	29	http://	?a=7o7j2rirq0Wm%2BnEB7XwUeldLlwGZ6...	HTTP/1.1	200

```

"apkSha1": "57b273c2ed9dd3c04092c4263900756543b28148",
"apkSha256": "026647955268ec5bc1ffa4c2e1827f10b201afa0bb9a44fac4f99404738cb945",
"appId": "com.balloon.shooter.play",
"appName": "Balloon Shooter",
"appid": "com.balloon.shooter.play",
"applicationName": "com.ball.App",
"appname": "Balloon Shooter",
"ashas": "",
"category": 0,
"flags": 954777156,
      : "12611a9cd",
"md5": "60ab36e54a85d52e4fad7c0f35638a2e",
"permissions":
"android.permission.VIBRATE,android.permission.INTERNET,android.permission.ACCESS_NETWORK_STATE,com.google.android.finsky.permission.BIND_GET_INSTALL_REFERRER_SERVICE"

```

Response from C2 server with spoofing instructions

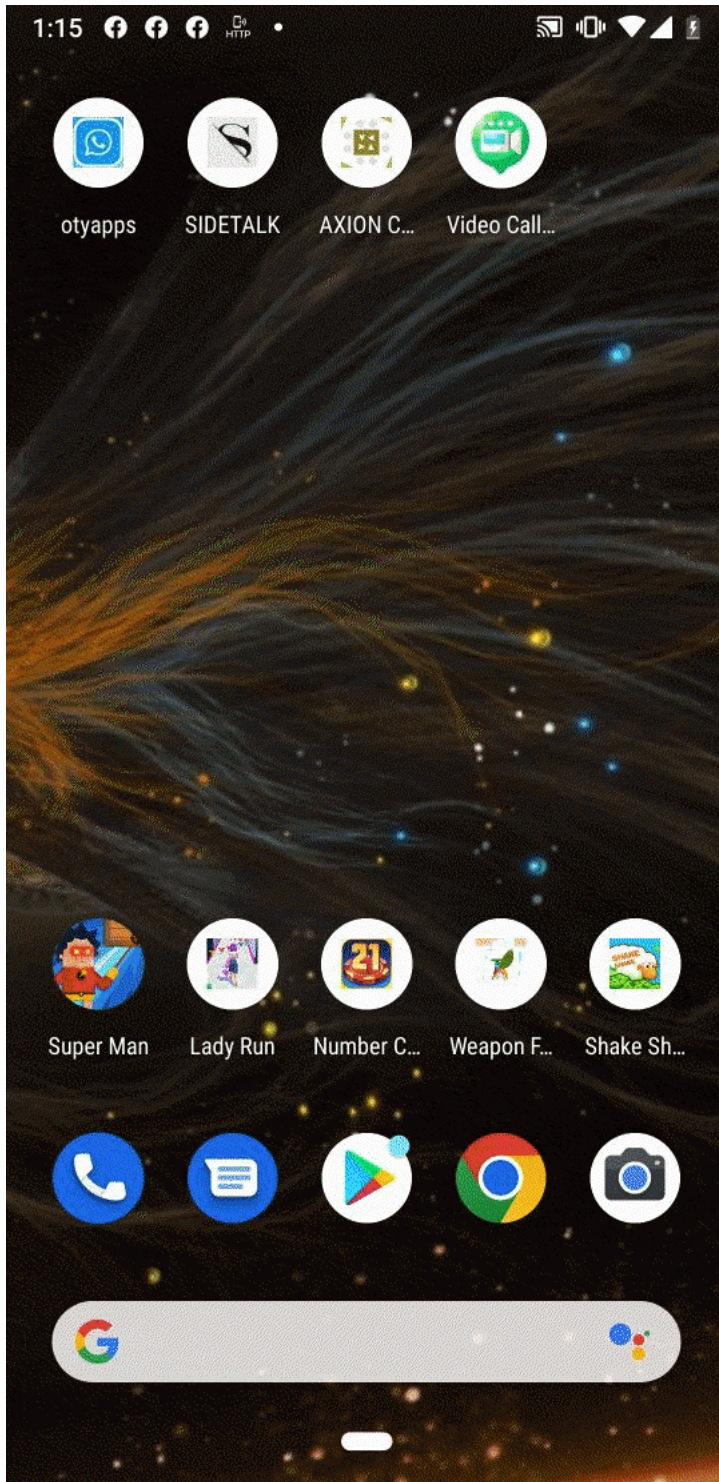
Source: Satori Threat Intelligence and Research Team

Here is a decrypted snippet of the C2 server's response. The app, in this case, is being told to mimic the app ID of a game called Balloon Shooter.

OOO/Hidden Ads

Many mobile apps show an ad as part of their loading screen - it's a common enough mechanism that most of us wouldn't think twice about seeing an ad when opening an app.

What's not common, however, is for the app to tell advertising platforms that it has displayed an ad to the user without having ever *actually* done so.



Loading a Scylla-associated app

Source: Satori Threat Intelligence and Research Team

The above gif seems straightforward - a game is opened, there's a brief loading screen, and then the game is ready to play. However, what the phone user *does not* see is that multiple virtual screens (aka "webviews") are launched in a way hidden from the user. These webviews are loaded with ads, tricking advertisers into paying for fake impressions to an audience that is never there.

WebView in com.shake.earn.sheep.causalgame (97.0.4692.87) trace

about:blank about:blank
empty at (0, 1789)
inspect pause

Ads Banner data:text/html;base64,PGh0bWw+PGhYVWQ+PHRpdGxIPiVuaXR5IEFkcyBCYW5uZXI8L3RpdGxIPjxtZXRhIG5hbWU9InZpZX...
empty never-attached
inspect pause

about:blank about:blank
empty
inspect pause

about:blank about:blank
empty
inspect pause

about:blank about:blank
empty never-attached
inspect pause

https://
empty never-attached
inspect pause

about:blank about:blank
detached size 840 × 131
inspect pause

about:blank about:blank
empty never-attached
inspect pause

Ads WebView about:blank
empty never-attached
inspect pause

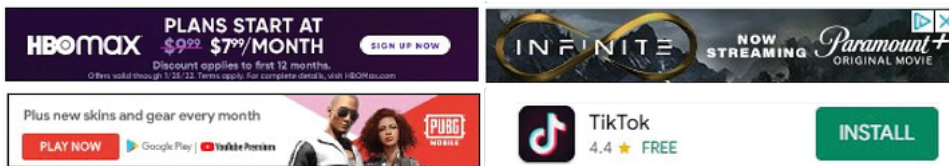
https://
empty never-attached
inspect pause

about:blank about:blank
empty never-attached
inspect pause

WebView of traffic of the Scylla-associated app

Source: Satori Threat Intelligence and Research Team

As you can see in the above log, several windows rendered...but not where the user could see them. Here is a sample of the actual ads which were "loaded" to the off screen WebViews:



Source: Satori Threat Intelligence and Research Team

Fake Clicks Using Receiver Code

Earlier in this post, we described the purpose of Receivers within apps: they're helpful in determining when a device is being actively used (or not) and when a device is well-suited for an ad to run (or not). Scylla's designers abuse these device functions to find the ideal times to "serve" ads when they *won't* actually be seen. To fully understand how they accomplished this, we have to start with the JobScheduler.

```

private void qc() {
    JobScheduler v0 = (JobScheduler)this.getSystemService("jobscheduler");
    if(v0 == null) {
        return;
    }

    v0.cancelAll();
    JobInfo.Builder v0_1 = new JobInfo.Builder(0x400, new ComponentName(this.getPackageName(), Afw.class.getName()));
    v0_1.setPeriodic(2000L);
    v0_1.setPersisted(true);
    v0_1.setRequiredNetworkType(1);
}

```

View of JobScheduler within a Scylla-associated app

Source: Satori Threat Intelligence and Research Team

JobScheduler *schedules a job* for Android apps or devices, as the name might suggest. In the Scylla operation, JobScheduler tells the app to run certain activities (like ads) when the correct conditions are met. To figure out if those conditions are met, the apps use Receivers.

```

private void qc() {
    JobScheduler v0 = (JobScheduler)this.getSystemService("jobscheduler");
    if(v0 == null) {
        return;
    }

    v0.cancelAll();
    JobInfo.Builder v0_1 = new JobInfo.Builder(0x400, new ComponentName(this.getPackageName(), Afw.class.getName()));
    v0_1.setPeriodic(2000L);
    v0_1.setPersisted(true);
    v0_1.setRequiredNetworkType(1);
}

```

JobScheduler triggering on on/off/user_present within a Scylla-associated app

Source: Satori Threat Intelligence and Research Team

Above, receivers determine if a device's screen is on or off and whether the device is actively in use. These determinations feed into the apps' logic about when and where to show ads.

```

00 public class C6121i {
    public static Activity ALLATORIXDEMO;

    /* renamed from: Ta */
01 public static void m16530Ta(Activity activity) {
02     if (activity != null) {
73         new FlurryAgent.Builder().withLogEnabled(true).build(activity, x.m15830k("k3xCEj}2\u000b&~9\u000f*
28         ALLATORIXDEMO = activity;
77         C6125g.m16537s('');
77         C6125g.m16538u(ALLATORIXDEMO, new C6140p());

    /* renamed from: wa */
    public static void m16592wa(Activity activity, String str) {
        ALLATORIXDEMO = activity;
        Mask.getInstance().setRewardedVideoListener(new C6139o(activity));
        Mask.getInstance().setAdsLaunchedListener($Lambda$x$faTA39rkjL30rVlxy3j57a1H0NA.INSTANCE);
        OneMask.getInstance().setAdsLaunchedListener($Lambda$x$GwU3gJSWaXmZUJzhGx3GbN_VVTI.INSTANCE);
        TwoMask.getInstance().setAdsLaunchedListener($Lambda$x$A_c2HsM5p_YknJYy8pUsMB6TvNU.INSTANCE);
        ThreeMask.getInstance().setAdsLaunchedListener($Lambda$x$Tn2xZit91rp3tnmFXG00x_9nUE.INSTANCE);
        Mask.getInstance().setInitializationListener(new t());
        Mask.getInstance().setHostName('');
        Mask.getInstance().init(activity, str);
    }
}

```

Listeners triggering a call to the C2 server

Source: Satori Threat Intelligence and Research Team

Much of what you see above is common to what the Satori team witnessed in the Poseidon operation. What's *new* in Scylla is how the apps fake actual *clicks* of the ads in question. Fake clicks can have multiple benefits for the fraudster: for ad networks that bill on a *views* model, clicks demonstrate effectiveness, which makes advertisers want to stick around. But some other ad networks bill by the *click*, which incentivizes the fraudster to just fake the clicks to get paid.


```

    }
    else if("top-right".equalsIgnoreCase(arg22)) {
        v8 = v3 - v10 * 2 - v13;
        v5_1 = v5[1] + v11 * 2 + v14;
    }
    else if("top-left".equalsIgnoreCase(arg22)) {
        v8 = v5[0] + v10 * 2 + v13;
        v5_1 = v5[1] + v11 * 2 + v14;
    }
    else if("bottom-right".equalsIgnoreCase(arg22)) {
        v5_2 = v3 - v10 * 2 - v13;
        v8 = v5_2;
        v5_1 = v4 - v11 * 2 - v14;
    }
    else {
        if("bottom-left".equalsIgnoreCase(arg22)) {
            v5_2 = v5[0] + v10 * 2 + v13;
            v8 = v5_2;
            v5_1 = v4 - v11 * 2 - v14;
            goto label_220;
        }
        v5_1 = 0;
        v8 = 0;
    }
}

label_220:
if(v8 <= 0 || v5_1 <= 0 || v8 >= v3 || v5_1 >= v4) {
    v8 = new Random().nextInt(v7 * 2) + v7;
    v5_1 = new Random().nextInt(v9) + v9 + 2;
}

v2[0] = v8;
v2[1] = v5_1;
DeviceLog.debug("this is ad click format = " + v6 + " videoType = " + v15 + " positionRandom = " + v12 + " closePosition = " + arg22 + " clickPosition = " + Arrays.toString(v2));
return v2;
}

private static int calculateTime(String arg8, String arg9) {
    try {
        String[] v8 = Devices.getConfigParams(arg8, arg9).split(",");
        String[] v3 = v8[0].split("-");
        String[] v8_1 = v8[1].split(",");
        if(v3.length == 5 && v8_1.length == 4) {
            int v0 = new Random().nextInt(1000);
            int v5;
            for(v5 = 0; v5 < v8_1.length; ++v5) {
                int v6 = v5 == 0 ? 0 : Integer.parseInt(v8_1[v5 - 1]);
                if(v0 >= v6 && v0 < Integer.parseInt(v8_1[v5])) {
                    int v8_2 = Integer.parseInt(v3[v5]) * 1000;
                    int v9 = Integer.parseInt(v3[v5 + 1]) * 1000;
                    return new Random().nextInt(v9 - v8_2) + v8_2;
                }
            }
            return 1000;
        }
    } catch(Exception unused_ex) {
    }
    return 1000 + new Random().nextInt(4000);
}

public static /* synthetic */ void m16486Ba(RewardedVideoAd rewardedVideoAd) {
    StringBuilder insert = new StringBuilder().insert(0, b.m16512k("YKFR\u000b\u0000ELEMdGICL\u0000rQ^ fBCEI"));
    insert.append(rewardedVideoAd);
    insert.append(C6143x.m16586k("\u000b\u0000ff\u0006b/f(\u000b8C"));
    insert.append(NiceUtils.getFbAdPkg(rewardedVideoAd));
    DeviceLog.error(insert.toString());
    VideoExTask videoExTask = new VideoExTask(rewardedVideoAd);
    if (Clicks.canMockClick()) {
        videoExTask.setClickData(Clicks.clickDelay(), Clicks.isClickReturn(), Clicks.returnDelay());
    }
    videoExTask.start();
}

```

```

public class FunUtil {
    public static final ExecutorService a;
    public static SoftReference b;
    public static String c;

    public static {
        FunUtil.a = Executors.newFixedThreadPool(1);
        new StringBuilder().insert(0, "installTime").append("FunUtil");
        FunUtil.c = "installTimeFunUtil";
        Context v0 = a0.a();
        if(v0 != null) {
            FunUtil.b = new SoftReference(v0);
            if(!a0.a(v0).contains(FunUtil.c)) {
                SimpleDateFormat v1 = new SimpleDateFormat("yyyy-MM-dd", Locale.CHINA);
                a0.a(v0).edit().putString(FunUtil.c, v1.format(new Date())).apply();
            }
        }
    }

    public static void init(Context arg2) {
        Application v2 = (Application)arg2.getApplicationContext();
        if(o.a(v2).getAll().size() < 20) {
            v2.registerActivityLifecycleCallbacks(new o());
        }
    }

    public static void onAdClick(String arg3, String arg4) {
        if(TextUtils.isEmpty(arg3)) {
            return;
        }

        Context v0 = (Context)FunUtil.b.get();
        if(v0 != null) {
            e0 v2 = new e0(v0, arg3, arg4);
            FunUtil.a.execute(v2);
        }
    }

    public static void onFbAdClick(Ad arg4) {
        String v4 = a0.a(arg4);
        if(TextUtils.isEmpty(v4)) {
            return;
        }

        Context v0 = (Context)FunUtil.b.get();
        if(v0 != null) {
            e0 v2 = new e0(v0, v4, a0.b);
            FunUtil.a.execute(v2);
        }
    }

    public static void setHostName(String arg0) {
        e0.f = arg0;
    }
}

```

Code within Scylla-associated apps log click locations and save them to later fake clicks

Source: Satori Threat Intelligence and Research Team

Above, the host apps track where *real* clicks happen on the device and store that information to later *fake* clicks.

But wait, there's more.

Obfuscation

As noted above, one of the other key differences between the original Poseidon operation and the latest Scylla operation is how well the threat actors manage to cover their tracks. Many of the screenshots above may be difficult to parse for the average observer because bits of the code have been "renamed" in an effort to make it harder for security researchers like our Satori team to reverse engineer.

```

private native void fakeApp(Application application);

private native void fakeDex(Context context);

static {
    System.loadLibrary("native-lib");
}

```

Above, code from the original Poseidon attack stage with an *unobfuscated* command. Below, a Scylla app with the same command, now *obfuscated* with Allatori.

Source: Satori Threat Intelligence and Research Team

```

public static Context ALLATORIxDEMO;

private native /* synthetic */ void fakeApp(Application application);

private native /* synthetic */ void fakeDex(Context context);

static {
    System.loadLibrary(a.C("V\\tK\\u0006N\\r\\b\\u0019\\u0006j"));
}

```

Additionally, the bad actors behind Scylla used an unpaid, demo version of Allatori:

```

public final class p implements k {
    @Override // com.laddd.i.k
    public void ALLATORIxDEMO(String str) {
        try {
            if (new JSONObject(str).getString(f.k("wb")).equals(f.k(""))) {
                x.wa(i.ALLATORIxDEMO, f.k("8c6c500724cc557c"));
            }
        } catch (JSONException e) {
            e.printStackTrace();
        }
    }
}

```

Example of use of Allatori code obfuscation tool, deobfuscated for investigation purposes.

Source: Satori Threat Intelligence and Research Team

Navigating Charybdis and Scylla

HUMAN's Satori team has worked closely with the Google Play Store and Apple App Store to ensure that all of the apps we've identified as being a part of the Scylla operation have been removed from public access. We've also worked closely with advertising SDK developers to mitigate the impact of the operation to their processes and their advertising partners. Our researchers will continue to monitor the threat actors behind the Poseidon, Charybdis, and Scylla operations for further adaptation.

Customers of [HUMAN's MediaGuard solution](#) are protected from fraud perpetrated by the Scylla operators.

Next Steps

HUMAN and the Satori team recommend the following:

For the public:

- If you have any of the apps listed in the below Indicators of Compromise section on your devices, remove them. Even though HUMAN is monitoring the threat actors, they may update the apps to change how they work, so removing the apps is your best bet.
- Consider reducing your risk of running dangerous apps by considering the source of the application and the trustworthiness of the brands involved. Secondary marketplaces, side-loaded or "cracked" applications may be much higher risk.

For HUMAN clients:

[MediaGuard](#) helps protect you from the impacts of Scylla-related fraud. Please contact your account manager with any questions about the impacts of Scylla to your organization.

For application developers:

The [app-ads.txt initiative](#) (PDF), spearheaded by the IAB Tech Lab, is a key step in protecting you and your applications' reputations from fraud through spoofing appIDs.

For advertisers:

Ensure you work closely with supply-side partners whose SDKs follow the [OMID protocol](#) for identifying ad-serving apps.

Indicators of Compromise

The below is a list of all of the apps associated with each of the Poseidon family of operations.

Scylla:

OS	sha256 hash	Name	Package
iOS	n/a	Loot the Castle	com.loot.rcastle.fight.battle (id160263)
iOS	n/a	Run Bridge	com.run.bridge.race (id1584737005)
iOS	n/a	Shinning Gun	com.shinning.gun.ios (id1588037078)

iOS	n/a	Racing Legend 3D	com.racing.legend.like (id158957945)
iOS	n/a	Rope Runner	com.rope.runner.family (id16149877C)
iOS	n/a	Wood Sculptor	com.wood.sculptor.cutter (id1603211)
iOS	n/a	Fire-Wall	com.fire.wall.poptit (id1540542924)
iOS	n/a	Ninja Critical Hit	wger.ninjacriticalhit.ios (id151405540)
iOS	n/a	n/a	com.TonyRuns.game (n/a)
Android	a7fea60c806d2dd10482cd630834373e49b4d944aa19d47091b3bd0a9d7890be	Super Hero-Save the world!	com.asuper.man.playmilk
Android	f90565a625044883ec9ce6323e813420412cf5c11e5f48f111dec4e1664cd176	Arrow Coins	com.arrow.coins.funny
Android	94d10d5839343072aa41bdd61259b580e567e621818686559dffe888dd9337eb	Parking Master	com.ekfnv.docjfltc.parking.master
Android	d43ee224e32e7d5d4ab901810afd7148cbf20ad57b07dc193cd154078ce433bb	Lady Run	com.lady.dress.run.sexylady
Android	0fc35fba6a708d5c6ca7da13e294cab70530c63828d9e4dbadf430e5c8f65b4b	Magic Brush 3D	com.magic.brush.gamesly
Android	0efc7c822685ebdf8f4d3d7d3104a957059c432b80624e3de6602c41bc95347f	Shake Shake Sheep	com.shake.earn.sheep.causalgame
Android	f0693787bffcde05fe419518621f199765ee802e00be50102c380563db678e9b	Number Combination: Colored Chips	com.yigegame.jyfsmnq.gg
Android	fe1d3f3c2ed8858c0327559f66149a4f0b82b01693b296e64448dfcff9d24ae0	Jackpot Scratcher-Win Real	com.physicswingsstudio.JackpotSca
Android	973ffa44c550c25b9ef80e176bb64f170647fd5ee115a3d03b98c88cc9a70a16	Scratch Carnival	com.scratchers.jackpot.luckypiggy
Android	355a9622553ddb7bcc78a2a04ce2f5c6067ac07e8f31168e76b884c2403923f	Ztime: Earn cash rewards easily	com.pocky.ztime
Android	4e94ab4fddde2f3f891777d21e4859aaa68dc9366aca82e15418692afb91af36	Billionaire Scratch	com.free.tickets.scratchers.Billionaire Scratch
Android	438e097fa5796400141f6b88b430829c2e3eebda7531e2deec4cf81d0ca15404	Lucky Wings - Lotto Scratchers	com.free.scratchers.luckywings
Android	420822558735922aee5efd5da2c22ffade14597e3f1d11ebf11ab12f0b3791b7	Lucky Star: Lotto Scratch	com.free.tickets.scratchers.LuckyLott

Android	3b1210ec0bd143b3131a04ac237d00fa6c1639a4679f2ebd694132145ce65f2	Shake Shake Pig	com.idle.merge.free.coinspiggy
Android	6ad0de5604b1f787908fc277780b4d45dbcbfcd98304b1f4ecf61c12e0fcc95	Lucky Money Tree	com.idle.merge.lucky.moneytree
Android	0a201d6da8e7a1b167bda949634d00d389dc995776bad398d0272c27cf85e710	Run And Dance	com.tap.run.and.dance
Android	539556f6ec50a7be633a1b5c3e30f01ba6520d6c28b1c6a57418636118e7677c	Lucky Scratchers: Lotto Card	com.lotto.bingo.lucky.scratchcard
Android	54c76607362e52798ae7535e8f7ead83474ecb49944e4bfc3f29537eb2e4a40c	Pull Worm	com.pull.bugs.worm
Android	ca0518bb9e77c498a222fdd0d99afc4b56eaa7ed2cf7953711d2a5872cf4e689	Crowd Battle: Fight the bad guys	com.crowd.battle.goamy
Android	ec47369e7557b695784ae4b7bb419cf6d2a4978b53d94f83b3ff50b0f24138db	Shoot Dummy - Win Rewards & Paypal Cash	com.shoot.dummy.fast.speed.linger
Android	4e9fd0291f921711fb19dd21d941a324c72cb77a23a9111c631304d192cea1ca	Spot 10 Differences	com.different.ten.spotgames
Android	2a4934fdb93b410d838f9b70f1d9a0b6a144016a5670cbf3b5df070273ff671f	Find 5 Differences - New	com.find.five.subtle.differences.spot.r
Android	3d67493bfd6eeadffef1d31f0eda3cb2ce11876f5a6458456322d536a0c9e7ff	Dinosaur Legend	com.huluwagames.dinosaur.legend.p
Android	8089bd929512552664daf8f28bee643bb88bf3ceadc60d0faa439072473f804	One Line Drawing	com.one.line.drawing.stroke.yuxi
Android	27d7f05cc02bcfe7b1d5e746c5025b7bffe2534086ca4f0c04336c2d12891283	Shoot Master	com.shooter.master.bullet.puzzle.hua
Android	4d1ccfa348c6c20b0291ac3ec2dacca47ed89b14f9d4b8ccde683f368494dc9f	Talent Trap - NEW	com.talent.trap.stop.all
Android	b6399d21f7a72f1b0bbc73864c9c228c5a4b8f6331aa5133d56589439f9747e7	Shoot it: Using Gun	com.bullet.shoot.fight.gtommm.tom
Android	d6fc218b1ca0eff4e2ee65dcd2a2b173192d9c0617de5134a150bbf6972e195c	Super Flake	com.chop.slice.flake2020
Android	0695f9028b1a6038ece961b494971f6762b403952c37b7f95e66112d8fc8027b	Five-Star Slice	com.five.star.slice
Android	1cea4435ead7d166e509f3cc9c9a556383822ca98ee3fdfd48c4c9d683f833d3	Sand Drawing	com.sand.drawing.newfight
Android	21c77771d23805593ff3f637245992376d7dce6fc18de4862973b94c184aa91c	Mr Dinosaur: Play your Dino	com.topgame.facego.finger.crazy.di
Android	abb3ae69fe1f6cf35b54508025c8a6734fe35ded22a6d42bcbcc65aa75bdbb3b	Track Sliding New	com.track3d.sliding.new

Android	08aeda329f3e0807625f228db30001ef83ff48fa82962bc50f84f8904a20dafb	Beat Kicker New	com.beat.kicker.two.game
Android	d51080d79668b6d4036a7e5ece86389f7f12305b8ca2c379559b764455406180	Fill Color 3D	com.cube.fill.color.paint.turn.fe
Android	8d5f8dd171fff8dd838871a519ae243c5bb4beeaf904cd365b97a05506aefb8a	Draw Live	com.draw.live.milipop
Android	7bc9e4ebf7ea018d87ac1fb47bdebe3dcf94dd02d0d8fefef902c22166d33493	Draw 1 Stroke	com.draw.one.line.stroke.xipi
Android	53739fa52192f8493da4a0067ec27c753a017941325f913508a8c8c840b534d9	Fidget Cubes	com.fidget.cubes.feel.like
Android	4cb13dbb442a0b2c657e5c79a60e856f86e18ee425896b996b9a6452519aca2c	Girls Fight	com.girls.fight.fly
Android	ee99a02f917a083348984b29bbfec81ec545bc16d401b391c923db14d546e353	Ninja Assassin	com.knifeninja.assassin.dltc
Android	69aff645ee74bb7fa0ec4c142a7c5b07da768742051a05d41ecb5ac456d5d28	Shooting Puzzle 2020	com.my.bullet.shooting.man.hunter.y
Android	1a23244555d1dbd9ead7019ed07dc6af157e0321e24182c53329be5d734a49ca	Pulley Parkour	com.pul.parkour.bbroller
Android	18ad35cadd4a4a15a40a85ef577645b0ff70199b11b25732e94b403f0063bf23	Chop Flake 3D	com.slice.chop.superslice3d
Android	e8d7baecb47b3a671c99cb0be21cb6b9e3ca4513cff6f02e51ac9d417d88b73b	Weapon Fantasy	com.weapon.fantasy.games
Android	90857e546fc916b77b1c307ad5a8c19ab213bdadd201350d4c7e6a418bde489f	Balloon Shooter	com.balloon.shooter.play
Android	62272f387631a5dc4b5561ad892170f94cb1ebf804a0c863929dae76084e9529	Musical Shoot	com.ltcmusical.fun2021
Android	f0d976fd5061ca5e5a8982a7f7cfafe472f6570a7de7f752501260599034c744	Chop Slices	com.lvdiao.chop.slices.chef
Android	c5c51f40f5bf4e63795cfd5d2e21ee64c1a586c00249a84ad315b2c7ea30acc	Ninja Slice	com.slice.masked.games
Android	c3db0ee64786b4aec5d61c3a993ab25fab2c04d2691ddfe4aae2e0fc87d4952f	Work Now!	com.work.now.slack
Android	c4a7057b987ce0daf8e355e801b2ab9386a72eaa80d3592dadfd4c55e5997304	Bottle Jump	com.bottle.jump.flip.challenge.fun
Android	2b109f567bcd83ab6548add5bbad392f0e5b4e7d3eeedd1363c46bdcc95d8ed3	Corn Scraper	com.corn.scraper.cut.pipe.siling
Android	b9c94a1c06e671600fd4d9dbfc2a7852a3786d10394afbf477f1b6df7c74952	Idle Wood Maker	com.idle.wood.maker.gametwo
Android	1a667ba74f9ec4b0d1d36408d8af1ff2493156960134178ff84f4a4d36837c15	Pop Girls Schooler	com.pop.girls.schooler
Android	2ce1c4dc30313facbdec12030cbd3a94dbc66dbfd2feab7bb231a3bb847b7a86	Romy Rush	com.romy.rushrun
Android	ccecf8310b7d300abe7ff808df4198ed4d0c9e734aaf8c95413d6a67abc4290c	Spear Hero	com.spear.super.man.hero

Android	33472b314c5beb8a0927522e9adf74e38b92ec5638f10d58457072177b926fb1	Dig Road Balls	com.dig.road.balls.play.games.ygyga
Android	97ed09697b85cf1d58aebc4d8f306c19beff1b949b80682c414f33d902521778	BOO Popstar	com.boostar.boo.popstar
Android	6687bdee0ecfec779f660327832d17739d95a4dd06536bbb387bd1175d5eb65c	Draw CompleteA	com.darwa.completea.ltca
Android	2f949069c0951b7d8203d17d401e0ad5b42e7bb3c0b8c56b4ac5383c3a21c336	Rush 2048 : 3D Shoot Cubes	com.rushcube.puzzle.block
Android	6c00ad91e11ded8b006372d95ab40f0d0769a2fbaaf21909b42bfd147fe74df	Meet Camera	com.magicvcam.hdmeet.cam008
Android	f7f55f024a5c3e5b18cde924ce25f72aab4523a148c24f0d9eea5d2581caa470	Auto Stamp Camera	com.stac.amper.qweaf
Android	028dfb91b3b54425ddb6445e2ec3ed01e1f5f736c6821447b67816148a3f6fe	n/a	com.find.five.differences.lvye.xsl
Android	d45b38198005b9de44994347e9d40ad822abaa969c3a53ec63cc31adcfb9772d	n/a	com.muvc.zwxfb
Android	67971ff706a897fdca3487bc5782a3813a3d3d2b4261524cc214f1f5b6ffdf27	Roll Turn	com.roll.turn.song.wusi.pt
Android	572cc347cbcdcbcf0f09470d22215f6791512ca0e74434e357c967e7515d9c2f8	Hiding Draw	com.hiding.drawltc.games
Android	fe8761e1be482bcea96d0edad74af19c6ce6ad85fe88178ff8343f0df61a1ccc	Peter Shoot	com.ltc.peter.shoot.tslgame
Android	e0c1774a83eda1420d5798a231592afdb8844ae7c0a2614142290d384327df63	Design n Road	com.ltcdesign.nroad
Android	5773e83f37ba16600f3e71bc5956d56be3a6c013e95f1c216bbf41eb8acf9be5	Draw Complete	com.ltcdraw.complete.fly
Android	0f1216681cca2781e73b50fbeb96bbe55a483aef79f2f5109e5776067a645fa4	Thief King	com.ltcking.thief.game.tsl
Android	96c35c5887b4a11a11590e9bba39771098ad83946eb0387cc52391b808191aaa	Downhill Race	com.downhill.race.redbull
Android	3c82a1c3fa04bbd4cb8b77f1ee5fe905b4a94d2e63f80a5ae19ecd71e9d0e920	Draw a War	com.draw.war.army
Android	b18072fb8f54e020e028cdd10fb5304ab55328c90c599fc2528d3e2b2713ccf8	Rescue Master	com.rescue.master.gear.mechanics.v
Android	e16dbfa2f712f8ccf36c07c5015142e0d63db0aec12d6efe656ce83ae285b8da	Spin:Letter Roll	come.letter.roll.race
Android	a65f2902fd3c067432804fd03b2c24de85af88a1f22596a1a3da65eb26cb716d	Helicopter Attack - NEW	com.helicopter.attack.shoot.sanba
Android	d40968b523ee99a78d752711e677bafa5a872f17d963640a4fac9418cca50ceb	Crush Car	com.crush.car.fly.delivery.lingjiu
Android	577f0644f9cd0849e51a6f5c0dadd7bafcc11b29b6353d97ffa039f3835747a6	Relx cash	com.tycmrelx.cash
Android	ec41dce767a5f49bbe1191f127f2f2c5a6a6345650b51a53795e273504062674	War in Painting	com.painting.war.inpaper

Android	17aa73a2c3967efd6b11b8c59cf51fcab839169b2281c7f435b2cca5e8ab1ab5	Bike Extreme Racing	com.bike.extreme.racing.bikegames
Android	576e5381142a2fe7fa8279d2a8d7fd4123adb7ae885029dca1bc5050ada2da0b	Player Spiral Maker 3D	com.player.spiral.maker.d3
Android	6d6a8f797415a26f51e1a5ba1de7c75cddd3b5a5a5c5148590355bf62f84b0be	Match 3 Tiles	com.blocks.tile.matching
Android	f9932711ca68d431824fa1557a333176e97faf529b1edde6bd7c0e8e2b614114	2048 Merge Cube - Win Cash	com.cube.merge.shooter

Charybdis:

App Name	App ID	Installs reported by Google's Play Store
[unknown]	com.car.jumpygy.race	0
[unknown]	com.droiand.killer.assassin.fun	0
[unknown]	com.ocean.fire.uabfire	0
[unknown]	com.pips.connect.uabking	0
[unknown]	com.recharge.go.run	0
[unknown]	com.Sign.Maker.cut.paint	0
Lucky Now! Scratch, Spin, Play Lottery & Win Money	com.lucky.scratch.spin.us	1,000,000
Lark - Work, Together	com.larksuite.suite	500,000
Assassin Legend - 2020 NEW	com.assassin.knife.legend.wuheng	100,000
Find the Differences - Puzzle Game	com.find.five.differences.picture.houpu	100,000
Wood Carving - NEW	com.wood.carving.cut.erqi	50,000
Idle Edo: Simulation of City Builder, Tycoon Games	com.edotime.picturescroll	1,000
Fresh Camera	com.cam.air.crush	1,000,000
Crush Car	com.crush.car.fly.delivery.lingjiu	1,000,000
Draw Color By Number	com.draw.color.number.paint.lvye	1,000,000
Happy Color By Number - New	com.happy.color.number.paint.hang.new	1,000,000
Stack Block Crusher	com.ygygame.stack.block.crusher	1,000,000
Disc Sport	com.disc.sports.battle.fly.yewai	500,000
Fire In The Desert	com.fire.the.desert.in	500,000

Running Dinosaur	com.games.mili.running.dinosaur.funny	500,000
Helicopter Attack - NEW	com.helicopter.attack.shoot.sanba	500,000
Help Me Down Game	com.help.me.game.puzzle.waocmangames	500,000
Rolling Scroll	com.rollscroll.rollingsc	500,000
Rugby Pass	com.rugby.pass.listgame	500,000
Color By Number	com.color.number.book.art.sanba	100,000
Find Hidden	com.find.hidden.objects.ltcltc	100,000
Flying Skateboard	com.flying.skate.rolling	100,000
Iron It	com.iron.clothes.smooth.perfect	100,000
Jump Jump	com.jump.ltcjump.game	100,000
Love Saver	com.make.love.saver.pop	100,000
Plant Monster	com.plant.eat.grow.monster.wuba	100,000
Sway Man	com.sway.man.escape.tumbler	100,000
Peter Shoot	com.ltc.peter.shoot.tslgame	50,000
Shooting Run	com.shoot.ing.rungame.ltcpp	50,000
Circuit Master	com.cir.cuit.master.likeapp	10,000
Desert Against	com.desert.against.ygygy.game	10,000
Balls Out Pazzle: Puzzle Maze Game	com.games.balls.out.puzzle.funny	10,000
Stacking Jump - Make Human Ladders	com.ladder.tower.stacking.jump.wuxiang	10,000
Musical Shoot	com.musicalltc.shoot.game	10,000
Bungee Jumper	com.bungee.jumper.gamelite	5,000
Rugby Master	com.master.rugby.discogame	5,000
ห่วยสามารถเบ็ด SamartBet ยี่เกี เกมส์ บาดา กีฬา	com.orghonoka.misamart	5,000
Magic and Throne	com.ylyy.magicandthrone	5,000
Draw & Puzzle	com.brain.guess.draw.puzzle.something	1,000
Color the Pictures	com.color.pictures.paint.number.yiliu	1,000
Crush King	com.crush.king.jump.truck.siliu	1,000

Happy Mouse!	com.happy.mice	1,000
King of Thieves	com.king.of.thief.easier	1,000
Magic Brain	com.magic.brain.think.solveologic	1,000
Props Rescue	com.props.rescue.girl	1,000
Find All	com.find.all.out.different	1
House Maker	com.bet.house.maker	0
Helicopter Attack	com.helicopter.attack.gun.rescue.liuliu	0

Poseidon:

com.magicvcam.beauty.yoobao.camera
com.onestoke.games.www.puzzlegame
com.time.date.stamp.camera.xixifunction
art.eff.filter.photo.editor
com.jelly.crush.europam.games
com.tools.blur.master.editor
com.mirth.cam.pic
com.mobwontools.pixel.blur.editor
com.blurcam.pro.usefultools
com.magicvcam.tool.beauty.camera
com.mobwontools.pixel.blur.cam
com.camera999.super.photo.editor
com.jelly.bubble.game.pop
com.pic.photo.editor.eraser
com.music.play.hi.cloud
com.puzzle.ygyline.onestroke
com.puzzle.lines1.drawstroke
com.video.nin.cut.face
com.camera.ygysuper.photograph
com.magicvcam.hdmeet.cam008
com.box.checkers.chess.free
com.super.photo.ygy.camera
com.magicvcam.meet.photograph
com.funny.camera.top
com.soon.ygy.photograph.camera
com.video.master.face.fb

com.magicvcam.camera.meet
com.camera.easy.photo.beauty
com.magicvcam.camera.newmeet
com.magicvcam.ygycronus.camera.magic
com.cos.ygy.camera.new
com.jelly.cube.crush.colors
com.photoeditor.background.change
com.connect.dots.maker.drawonelin
com.blur.oceans.flyin
com.magicvcam.mine.ygycamera.magic
com.ygy.find.spot.the.difference
com.candy.crushfunny.toystoys
com.frog.crushtoy.blast
com.magicvcam.ygycamera.magic.tool
com.toy.blast.cube.crush.toysfun
com.magicvcam.photos.ygy.tool.magic

[Previous Post](#)

[Next Post](#)