Rise of Banking Trojan Dropper in Google Play

zscaler.com/blogs/security-research/rise-banking-trojan-dropper-google-play-0

The Zscaler ThreatLabz team has recently discovered the Xenomorph banking trojan embedded in a Lifestyle app in the Google Play store. The app is “Todo: Day manager,” and has over 1,000 downloads. This is the latest in a disturbing string of hidden malware in the Google Play store: in the last 3 months, ThreatLabz has reported over 50+ apps resulting in 500k+ downloads, embedding such malware families as Joker, Harly, Coper, and Adfraud.

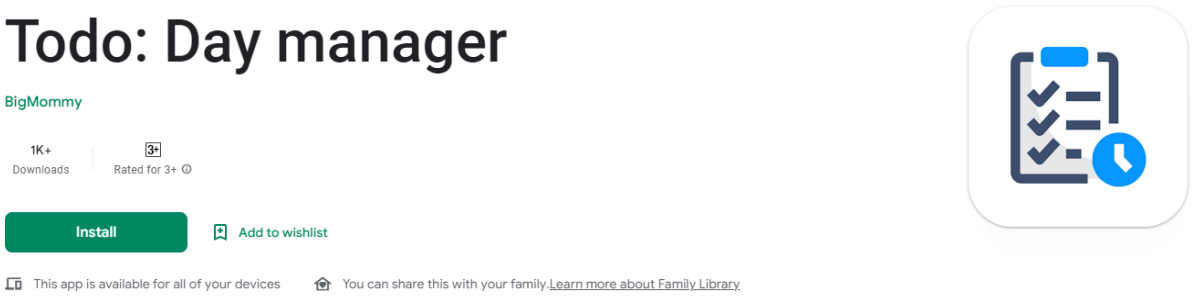

Fig no 1.Malware Installer From Play Store

Xenomorph is a trojan that steals credentials from banking applications on users’ devices. It is also capable of intercepting users’ SMS messages and notifications, enabling it to steal one-time passwords and multifactor authentication requests.

Our analysis found that the Xenomorph banking malware is dropped from GitHub as a fake Google Service application upon installation of the app. It starts with asking users to enable access permission. Once provided, it adds itself as a device admin and prevents users from disabling Device Admin, making it uninstallable from the phone. Xenomorph creates an overlay onto legit banking applications to trick users into entering their credentials.

A similar infection cycle was observed three months ago with the Coper banking trojan. This trojan was similarly embedded in apps on the Google Play store, and sourced its malware payload from the Github repo.

Technical Details

Below is the Xenomorph infection cycle once a user downloads an app and opens it.

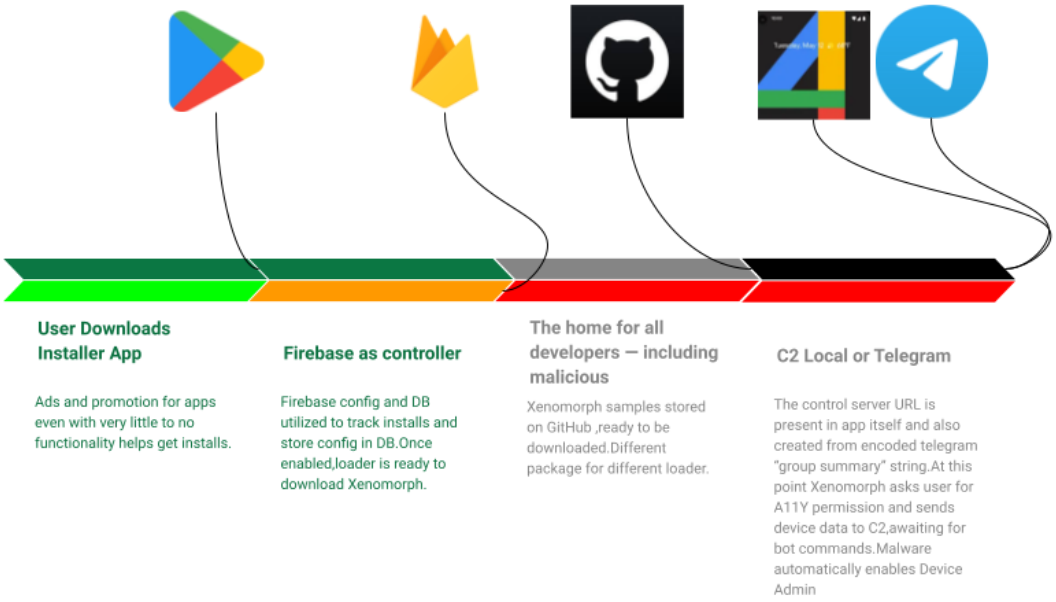

Fig no 2.Flow of infection

When the app is first opened, it reaches out to a Firebase server to get the stage/banking malware payload URL. It then downloads the malicious Xenomorph banking trojan samples from Github. This banking malware later reaches out to the command-and-control (C2) servers decoded either via Telegram page content or from a static code routine to request further commands, extending the infection.

The parent malware downloader (Google Play Store) application gets its config from Firebase for its database.

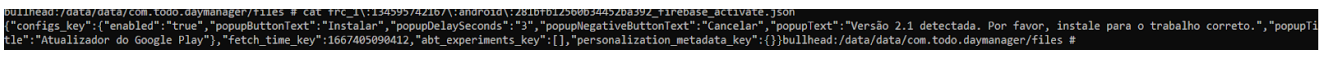

Fig no 3. Malware enables downloader.

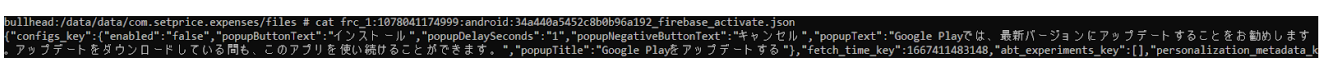

Fig no 4. Downloader not enabled.

As shown in the above screen shot, the malware will only download further banking payloads if the “Enabled” parameter is set to true.

The following screenshot shows how the Firebase database malware uses Github links to download Xenomorph payloads:

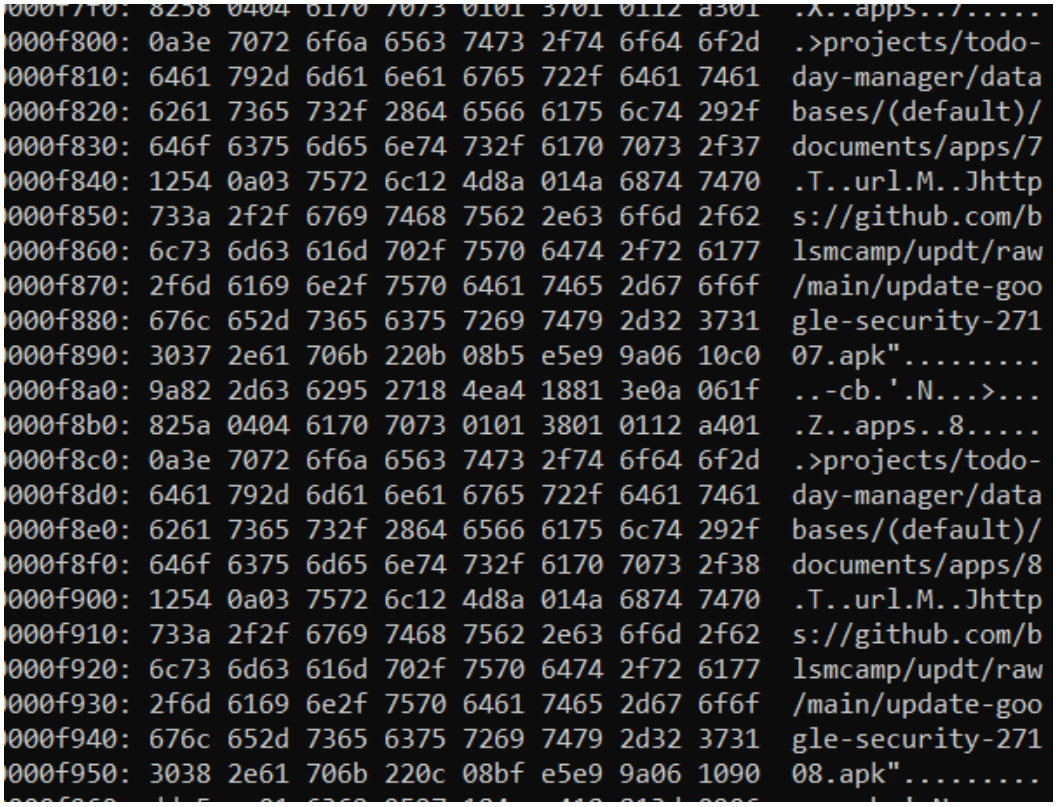

Fig no 5. The malware writes dropper URLs in local DB of firebase

The screenshots in Figures 6 and 7 below show the C2 retrieval from a Telegram page. Here the banking payload has the Telegram page link encoded with RC4 encryption. Upon execution, the banking payload will reach out to the Telegram page and download the content hosted on that page.

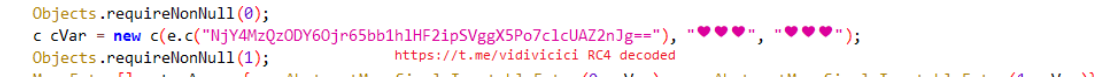

Fig no 6.Uses Telegram link response to create C2 in addition to static encrypted C2 present in app

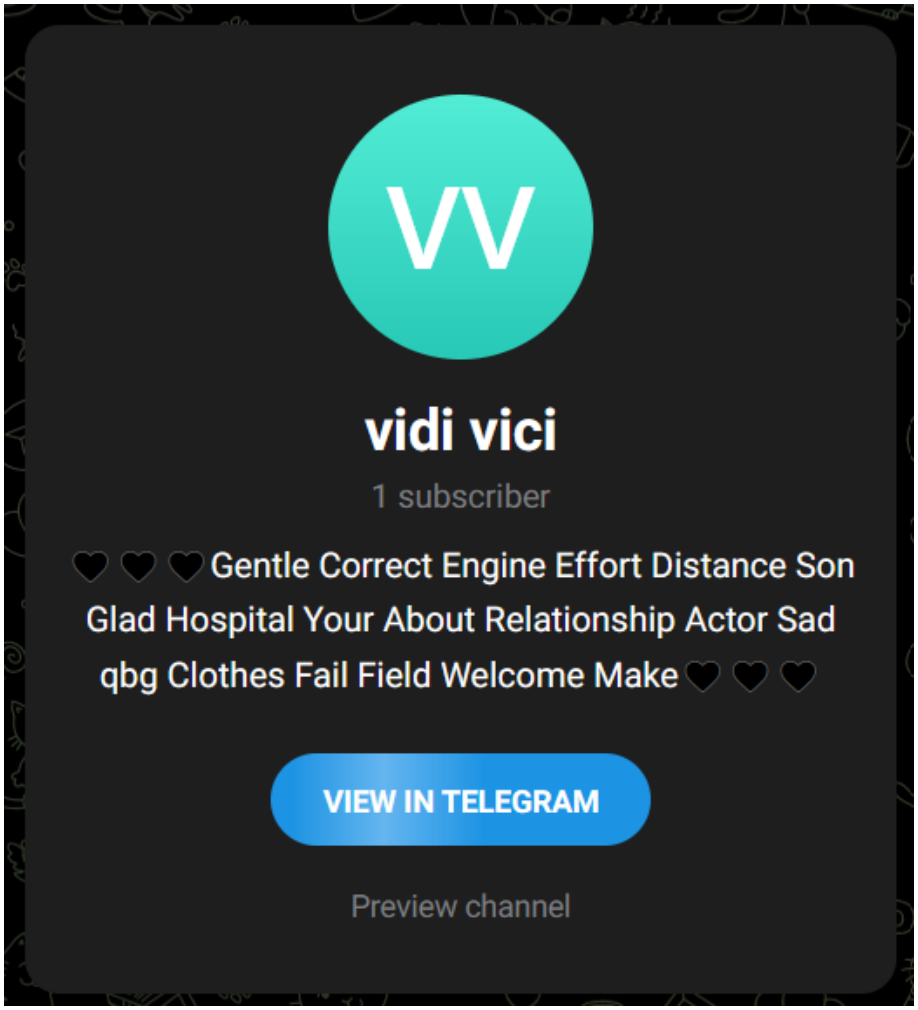

Fig no 7. Telegram channel preview where string in between hearts emoji is used to create C2

As per the following screenshot, the payload will decrypt the C2 server address from the downloaded content:

```
while (i3 != -1) {
    i3 = str.indexOf("❤❤❤", i3 + i4);
    if (i3 != -1 && (indexOf = str.indexOf("❤❤❤", (i2 = i3 + 6))) != -1) {
        String[] split = str.substring(i2, indexOf).split(" ");
        StringBuilder sb = new StringBuilder();
        int i5 = 0;
        for (String str2 : split) {
            Integer num2 = null;
            try {
                num = Integer.valueOf(Integer.parseInt(str2));
            } catch (NumberFormatException unused) {
                num = null;
            }
            if (num != null) {
                try {
                    num2 = Integer.valueOf(Integer.parseInt(str2));
                } catch (NumberFormatException unused2) {
                }
                sb.append(num2);
            } else {
                if (str2.length() == 3) {
                    char[] cArr = blglyu.rjqwgg.vveize.utilities.b.f77a;
                    String lowerCase = str2.toLowerCase();
                    StringBuilder sb2 = new StringBuilder();
                    for (int i6 = 0; i6 < lowerCase.length(); i6++) {
                        int indexOf2 = ("abcdefghijklmnopqrstuvwxyz".indexOf(lowerCase.charAt(i6)) - i5) % 26;
                        if (indexOf2 < 0) {
                            indexOf2 += 26;
                        }
                        sb2.append("abcdefghijklmnopqrstuvwxyz".charAt(indexOf2));
                    }
                    if (sb2.toString().equals("dot")) {
                        sb.append(".");
                    }
                }
                sb.append(str2.charAt(i5 % str2.length()));
            }
            i5++;
        }
        String validateAPI_C2_Telegram = sb.toString();
        if (!arrayList.contains(validateAPI_C2_Telegram)) {
            arrayList.add(validateAPI_C2_Telegram);
            this.f767a.add(new d(validateAPI_C2_Telegram));
        }
        i3 = indexOf + 6;
```

gogoanalytics.click

Fig no 8. Decode C2 from Telegram

ThreatLabz also observed RC4 encoded C2 domains stored inside the code. The following screenshot shows the C2 request in which the payload sends all the installed applications to C2 in order to receive further instructions. In one case, it will present the fake login page of a targeted banking application if the legitimate application is installed in the infected device.

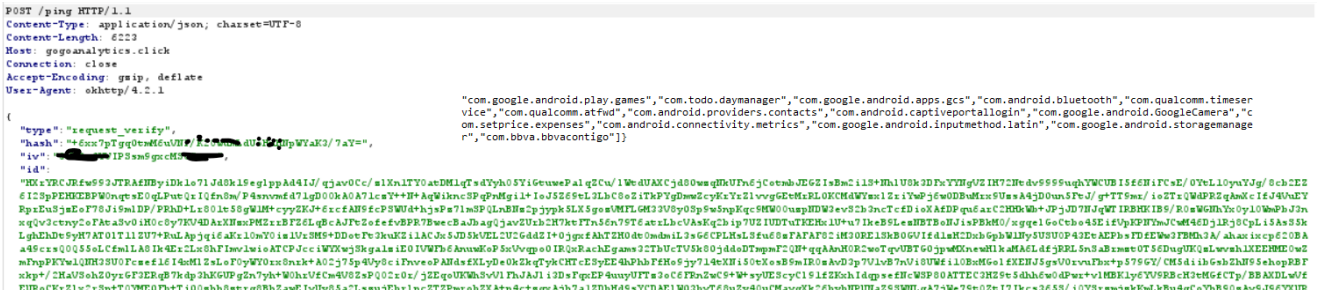

Fig no 9. Malware uploading all package information to receive commands

ThreatLabz also observed another application, named “経費キーパー” (Expense Keeper), exhibiting similar behavior. On execution of this application, it is observed that the “Enabled parameter” is set to false, same as the execution previously shown in Figure 4. Due to that, it was not possible to retrieve the Dropper URL for the banking payload. ThreatLabz is working with the Google Security team for the same.

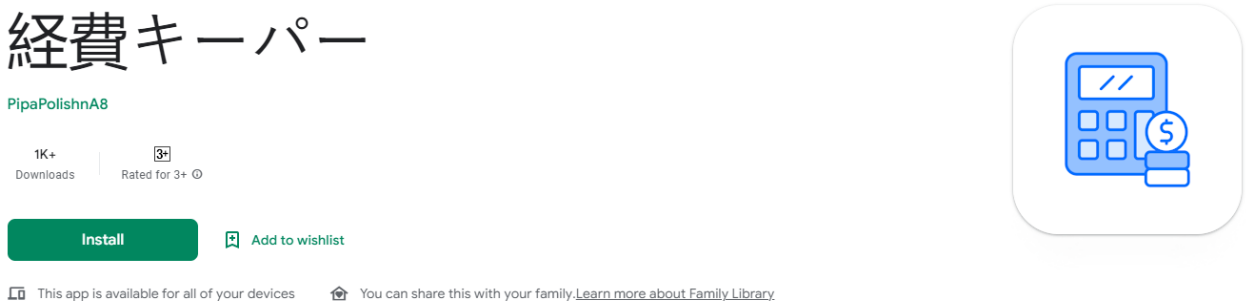

Fig no 10. Suspicious Installer exhibiting the same behavior

IoCs

com.todo.daymanager	d81f9c03c412b11df357f0878c9c5cad9319c7eea11b5c46d0c624995bc09563
com.setprice.expenses	58d634230951ee7699a4b4740e12be8e93a28bd183f61447832bd1d5d98160d8

Xenomorph banking trojan

Package Name	MD5
njuknf.cpvmqe.degjia	b8b8706807a97c40940109a93058c3d0
ylyove.pkmcsy.upvpta	98ea3fe61fde0c053dfac61977a11488
ylykau.jhfxjd.hlhhwl	df57895cfc79ee8812aac5756ab4bcc8
lkvrny.bbslie.mrgsdy	73511ef7bb9d59b3d91dbeef5f93eec0
gkapsv.nlitfn.fzteaf	f0b001dbe36f45cedcb15e3f9fc02fd7
binono.bgcwvl.iupqtk	8437e226e55ba6dea9a168bee5787b0d
cfbyzn.zhxxjj.sziece	8f66412e945ca9a75797d5f5eba9765c
gfgnfe.rcsjkm.abwxdj	6a117cafa32a680dc94f455745291f0f
usyjui.monkab.acacpn	cb9500f910bd655df444f7d43d0298f9

gnvbgm.ipblyp.bpnyrg	d95c03247a58d3fabb476a7f3241f3a1
xsgrsn.nicojr.uaqxws	cd63afae858fdf75f34aae05e36b8a34
xhlkae.ligagt.dmihjy	c5d510251a34f52427d133a6f9248cbf
qlvsvm.oqsncp.otgbxc	781bbaee614697beecfcbe9a2f9dd820
rxreyj.obxmlg.rjluib	49c4801abb6c92d17c8021c2f656c644
brpdxm.orolnd.jsxhrp	1829589d95bdd2c30f0bef154decd426
wwzaqw.eejyqr.czrldy	e834676cdbd63ce4eb613499605dc365
ogbfbt.rhrnua.kccuoh	9e498ba660bdcb279149e6a5986c2793
lnckvn.vlmjxx.uwcpub	4b2e849543b0ecaec1885170a5ef5243
vjqfyn.ygmzrs.trlvch	7e4f1deb5b21d47a7c41ef1a5f43a2f2
blglyu.rjqwgg.vveize	7f574986dc8a03e6a4cba60d1ac4f7d1

C2s

- hxxps[://]github[.]com/blsmcamp/updt
- gogoanalytics[.]click
- gogoanalytics[.]digital

Conclusion

At Zscaler we proactively detect and monitor such applications to secure our clients. Such bank phishing installers most of the time rely on tricking users to install malicious applications. Users are advised to keep an eye on what application is being installed. A Play Store application is not supposed to side load or ask users to install from unknown sources. We believe hostile phishing downloaders will further increase in prevalence in the future. User vigilance is of the utmost importance to defeat these phishing campaigns.