# Existing Firewall Solutions

The following Chapter will present and analyze existing Antivirus, Firewall and Security Solutions for Android and IPhone. Data for this comparison was gathered from descriptions of the app developer and through testing it. If features required further licensing, either a free testing period (if offered) or the feature description of the developer was used.



## Features

### Android

Most evaluated Android apps provide features beyond a simple Virus Scanner. They compete within the following categories: Antivirus, Firewall, Privacy, Backup, Antitheft and System-tweaks.

**Antivirus**Within the Antivirus category a scanner feature is provided by all Apps, which will scan your phone for Viruses and Malware. The algorithm as well as the signature database is executed and stored on the phone by the majority of Apps. However, the App from Sophos Ltd. does the processing in the cloud which obviously will produce network traffic. As far as described by the developer most of the scanner will check for virus and malware infections. Some companies choose to offer a malware scan as an optional premium feature, while others are very unspecific about what they scan for. While technically any virus is malware, news reports about viruses resulted into marketing Applications as Antivirus rather than Antimalware. In regard of the capability of these scanners, it all comes down to marketing which (falsely) splits the applications in Antivirus and Malware. While the underlying database with signatures is different the functionality is the same for both. Some apps allow the user to select the files, folders (incl. external SD cards), apps, etc. to be scanned. A scan scheduling feature is provided by some apps to allow the phone to execute the scanner automatically at the user defined date and time, which might also be sold as a premium feature by some Apps.

**Firewall**This feature category is divided into the classical firewall (blocking IPs and services) and what is referred to as “Web Shield” by most of the apps (blocking websites). Classical Firewall implementations does exist only in a minority of Apps, due to implementation restriction (as described in the “Technical” section). On the other hand all seem to provide the ability to filter websites.

**Privacy**Very few of the tested apps provide specialized scanners (sometimes as separate app) for facebook, twitter, or other social apps which will alert the user of insecure settings. The crucial features within the category are App-Permissions, Parental-Control (App Lock) and Spam Block. App Permission are presented to the user filtered by a scanner or as a classified list. Some offer to block permissions if root access is available. Parental-Control will block the execution of apps by asking for an unlock pattern or key. The spam blocking feature checks incoming phone calls and SMS and blocks them.

**Backup**While most apps provide the ability to store contacts, only a few also backup sms, call histories, file (pictures, documents,…). Application Backup is mostly available as a premium service (probably due to the traffic volume required).

**Anti-Theft**The features within this category try to offer security in case the device gets lost or is stolen. Most of the features can be triggered by SMS commands or using the developers web-service. Depending on the country (not allowed in Germany and Austria) the App will take a picture of anyone who enters the wrong unlock code more than three times. In case one has misplaced the phone, another feature will play a siren sound to locate it this way. However, the most advertised feature is locating via geo location (GPS, Network…). More advanced features, not provided by all apps include, remote controlling the security app itself, taking pictures, sending messages and initiate callbacks. One app even offers a geo fencing feature, where the geo location is used to block apps, content, or send automated alerts (calls, SMS, E-Mail) to user-defined contacts.

**System Tweaks**The simplest of these features display statistics, like network data usage, battery consumption and memory allocation. More advanced features allow to automatically stop running application or free memory by scanning for unused files.

### IOS

Compared to Android, very little Apps exist that handle security related Tasks. Those that does exist can be placed in the same categories as Android, but competition is very category specific.

**Antivirus**Unlike Android there are only a few Apps that advertise the use an antivirus scanner. Most of these scanners work in regard to known vulnerabilities, watch for unknown processes and malware or conduct cloud-based scans of cloud-stored files. Scan scheduling or other features available under Android are missing entirely.

**Firewall**Like with the Antivirus category, classical firewalls do not exist for iOS. There are a variety of Apps providing web traffic filtering, link on Android.

**Privacy**These kind of features are also missing from security apps, but are available separately for all kind of social media. Some apps offer Parental-Control settings as premium feature.

**Anti-Theft**Like the other categories, this functionality is provided by separate Apps which were not part of this analysis.

**Backup and System-Tweaks**Backup is limited to contacts and photos, and also system tweaks could only be found in one of the test apps.

## Technical Analysis

This chapter will analyze the previously introduced features on a technical level. It will try to explain how the firewall and web filter features as well as other features of interest are implemented on the Android and iOS platform.

### Android

**Antivirus and Backup**The exact inner workings of these features are hard to figure out without reverse engineering the software itself. The best guess is, that a java or c++ implementation of a signature matching algorithm is used to check all file made available by the operating system. Under Android this is usually a dedicated section of the internal flash, which is mounted to allow the user to store its data, or an SD-card. Files belonging to the Android System are not accessible without rooting the device.

**Anti-Theft and Systemtweaks**The features of these two categories are part of the Android system itself. By installing a service adapter (like in iOS) Android allows Apps to register as Device Administrator and take over remote management. An API then provides locating, siren, lock, … capabilities.

*The kind of anti-theft features that are available from the API have to be researched further, especially for unusual features like remote install/uninstall, remote callback, … provided by AVAST. Also further analysis should be conducted on how the SMS command works. The SMS commands most likely have to be implemented inside the App and may only work in Android devices up to 4.4.*

System-Tweak features are just refurbishing what Android already reports in some of the “Settings” Menu. These kind of information are already provided by the OS but are rather hard to find.

*Features like the Task-Killer are available, question is to which extend. Is an application allowed to stop another app from running, especially if that app didn’t start it in the first place? What kind of functionality is provided by the API and what kind of limitations exist?*

**Firewall and Privacy**Features in this category include a classic firewall implementation, that means blocking network traffic on a OSI-Layer 3 basis for the entire system. Such a feature is not possible without rooting the Android phone. Alternatively Apps offer what is advertised as Web-Shield and works in the same way AdBlocker features work. Not the web traffic itself is analyzed but the URL requested. This is done by calling API methods (see example code below) offered by the system-integrated browser or chrome. Other browsers are not supported.

@Override  
public void onCreate(Bundle savedInstanceState) {  
  // ...  
  if (Intent.ACTION\_VIEW.equals(action)) {  
    [String](http://www.google.com/search?hl=en&q=allinurl%3Astring+java.sun.com&btnI=I%27m%20Feeling%20Lucky) link = intent.getData().getScheme()  
      + intent.getData().getSchemeSpecificPart();  
    // method to handle the link  
    this.handleThisLink(link);  
  }  
  // ...  
}

*To which degree this filtering can occur, URL request or http packet content, shut to be researched further. The following links provide some further insight.*

public class VpnServiceEx extends VpnService implements Handler.Callback, Runnable {

private static final String TAG = "VpnServiceEx";

private Handler mHandler;

private Thread mThread;

private ParcelFileDescriptor mInterface;

@Override

public int onStartCommand(Intent intent, int flags, int startId) {

// The handler is only used to show messages.

if (mHandler == null) {

mHandler = new Handler(this);

}

// Stop the previous session by interrupting the thread.

if (mThread != null) {

mThread.interrupt();

}

// Start a new session by creating a new thread.

mThread = new Thread(this, "TestVpnThread");

mThread.start();

return START\_STICKY;

}

@Override

public void onDestroy() {

if (mThread != null) {

mThread.interrupt();

}

}

@Override

public boolean handleMessage(Message message) {

if (message != null) {

Toast.makeText(this, message.what, Toast.LENGTH\_SHORT).show();

}

return true;

}

@Override

public synchronized void run() {

Log.i(TAG,"running vpnService");

try {

runVpnConnection();

} catch (Exception e) {

e.printStackTrace();

//Log.e(TAG, "Got " + e.toString());

} finally {

try {

mInterface.close();

} catch (Exception e) {

// ignore

}

mInterface = null;

mHandler.sendEmptyMessage(R.string.disconnected);

Log.i(TAG, "Exiting");

}

}

private void runVpnConnection() {

configure();

FileInputStream in = new FileInputStream(mInterface.getFileDescriptor());

FileOutputStream out = new FileOutputStream(mInterface.getFileDescriptor());

// Allocate the buffer for a single packet.

ByteBuffer packet = ByteBuffer.allocate(32767);

Socket tcpSocket = new Socket();

boolean ok = true;

// We keep forwarding packets till something goes wrong.

while (ok) {

// Assume that we did not make any progress in this iteration.

try {

// Read the outgoing packet from the input stream.

int length = in.read(packet.array());

if (length > 0) {

Log.i(TAG,"packet received");

packet.limit(length);

String serverIP = getDestinationIP(packet);

int port = getDestinationPort(packet, getHeaderLength(packet));

Log.d(TAG, "destination IP: " + serverIP + " port: " + port);

InetAddress serverAddr = InetAddress.getByName(serverIP);

SocketAddress socketadd= new InetSocketAddress(serverAddr, port);

tcpSocket.connect(socketadd); \*\*\*\*\*// this fails\*\*\*\*\*\*

OutputStream outBuffer = tcpSocket.getOutputStream();

outBuffer.write(packet.array());

outBuffer.flush();

outBuffer.close();

packet.clear();

ok = false;

}

if (tcpSocket.isConnected()) {

InputStream inBuffer = tcpSocket.getInputStream();

byte[] bufferOutput = new byte[32767];

inBuffer.read(bufferOutput);

if (bufferOutput.length > 0) {

String recPacketString = new String(bufferOutput,0,bufferOutput.length,"UTF-8");

Log.d(TAG , "recPacketString : " + recPacketString);

out.write(bufferOutput);

}

inBuffer.close();

}

} catch (IOException e) {

// TODO Auto-generated catch block

e.printStackTrace();

Log.e(TAG,"exception: " + e.toString());

ok = false;

}

}

}

private void configure() {

// If the old interface has exactly the same parameters, use it!

if (mInterface != null) {

Log.i(TAG, "Using the previous interface");

return;

}

// Configure a builder while parsing the parameters.

Builder builder = new Builder();

builder.setMtu(1500);

builder.addAddress(getLocalIpAddress(), 24);

builder.addRoute("0.0.0.0", 0); // to intercept packets

try {

mInterface.close();

} catch (Exception e) {

// ignore

}

mInterface = builder.establish();

}

http://eridem.net/android-tip-001-intercepting-url-from-default-browser/  
http://stackoverflow.com/questions/4780899/intercept-and-override-http-requests-from-webview  
http://jerkyapp.com/blog/ad-block  
https://developer.appcelerator.com/question/147466/how-to-intercept-urls-from-browser

***VPN***  
Traffic of all apps is not accessible without root in the phone. There is a workaround, which allows to use the tunnel of the VPN service locally (without connecting to an external VPN server!) and capture traffic from other applications. However, as the following links show, there are some problems with forwarding the traffic (hence filtering).   
http://stackoverflow.com/questions/21399027/android-vpnservice-how-to-forward-intercepted-internet-traffic  
<http://stackoverflow.com/questions/13233477/how-to-use-vpn-in-android/#answers>

In later Versions of Android (> 4.2) also allow for multipath TCP, which uses all available internet connections without asking the user. *How this impacts the VPN solution (locally or external) has to be determined.*

### IOS

**Antivirus and Backup**Due to the strict development process Apple enforces there is little need for Antiviral or Malware functionality. *Existing solutions seem to scan the running processes and report possible vulnerabilities based on version numbers, etc.?*

**Anti-Theft and Systemtweaks**Decent system is already part of IOS but (like Android) may be taken over by a registered service.  
*Has to be determined how that is done exactly.*

**Firewall and Privacy**Like on Android the classical filtering Firewall is not available. *If jailbreaking the phone will overcome that limitation has be researched.*

***VPN***  
Traffic interception and filtering cannot be conducted locally. Available solutions do that on the VPN Server site. *IOS supports multipath TCP which allows the phone to use all available internet connections with asking the user. It has to be determined how this affects a VPN tunnel? Are packets in this kind of setup only send via the tunnel?!*

## Conclusion

While there are a huge amount of Applications available for Android, only a few and with with limited functionality exist for the IPhone. A reason for this are the development policies. While Google is fairly unrestrictive for releases to its web store, Apple is rigorously checking the behavior and source code before releasing it. This makes the IPhone a very secure platform with regard to virus and malware infections. Furthermore, neither allow for analysis of the entire internet data-stream (wifi or gms) without rooting or jailbreaking the device.

What both have in common is the ability to create VPN connections, where the entire traffic can be monitored and filtered on the server side (a local workaround exists for Android only).