CAcert system trusted certificates (without lockscreen)

The existing method of importing user certificates works fine, but it has the disadvantage that it requires a PIN / password lockscreen whenever user certificates are installed. By installing the CAcert certificates as system certificates, these files are better protected from tampering by malicious apps, and there is no lockscreen requirement (allows 'Slide to unlock' or no lock at all). You will need a rooted phone (or at least temporary root access), and a system with openssl software for creating the new certificates.

The next steps will show you how to create Android compatible certificate files from the original CAcert certificate files, how to install/import them on your android device, and how to verify everything is correctly installed.

It is possible, in Android OS version 4.4.2, to save certificates as user trusted ones (Android itself creates their correct names, derived from hashes), and then move them into the system trusted certs repository, using program for Android as Terminal, adb shell, or Ghost Commander. If you decide to follow this process, skip to the Importing paragraph replacing the source folder „/sdcard“ used there, with the „/data/misc/keychain/cacerts-added“ folder, where Android stores user trusted certificates. Do not copy CAcert roots, move them!

Creating

We will create Android compatible certificate files from the original CAcert certificate files. Get the CAcert root certificates from the cacert.org website https://www.cacert.org/index.php?id=3 Download the root certificate PEM format (root.crt) and the Class 3 PKI key in PEM format (class3.crt) Get the hash of the root.crt certificate:

openssl x509 -inform PEM -subject_hash_old -in root.crt | head -1

This shows you the hash, in the case of the CAcert PEM file 'root.crt' it is '5ed36f99' (note the use of '-subject_hash_old' instead of '-subject_hash', to get an openssl 0.9 compatible hash) We will use this hash value, append '.0' (dot zero) and use this as the filename for the resulting Android certificate:

cat root.crt > 5ed36f99.0

openssl x509 -inform PEM -text -in root.crt -out /dev/null » 5ed36f99.0

Repeat these steps for the Class 3 PEM certificate file 'class3.crt'. If things go well you will end up with the files 5ed36f99.0 and e5662767.0 (if you get the hash values 590d426f and 99d0fa06, you are not using the '-subject_hash_old' parameter to openssl). The md5sum of the certificate files:

md5sum 5ed36f99.0 05e5fcd7af6ba52e254d065b734213ab 5ed36f99.0

md5sum e5662767.0 000e8e995568091e1d411ff6deb4c118 e5662767.0

Importing

We now have Android compatible certificate files, and we will import them into Android 'System' certificate store. It is necessary for you to gain the super-user rights to be able to write to / remove from / move between system subfolders. To achieve this, the Android system has to contain the „su“ (super-user) program, which provides you with the super-user rights. Some phones' Android systems do not include this program. In such a case, you have to store all certificates added as the user ones.

Copy the files to the /sdcard folder, either with any file manager or with adb push. Go into adb shell (adb shell from commandline), or open the 'terminal'-application on your android device. You will get a command prompt similar like shell@android:/ $ Gain superuser/root rights, neccessary to perform privileged actions:

su

Make the /system folder writable (will return to read-only upon reboot):

mount -o remount,rw /system

Copy the new certificate files to the correct folder on your Android device:

cp /sdcard/5ed36f99.0 /system/etc/security/cacerts/ cp /sdcard/e5662767.0 /system/etc/security/cacerts/

Correct the file permissions to u=rw, g=r, o=r:

cd /system/etc/security/cacerts/ chmod 644 5ed36f99.0 chmod 644 e5662767.0

Check if the files are ok:

ls -al -Z

Omit '-Z' if you are using a version of Android without SElinux, it just shows some extra security settings which might be useful if you run into trouble.

Amongst the other default android certificate files, you will see the two new files:

-rw-r–r– root root u:object_r:system_file:s0 5ed36f99.0 -rw-r–r– root root u:object_r:system_file:s0 e5662767.0

The certificates will be loaded upon the next boot of your device, so reboot your device:

reboot

Verifying

To verify certificates are installed correctly, go to Settings → Security → Certificates. It should list both „CAcert Inc.“ and „Root CA“ among the other certificates in the 'System' section. Make sure that these CAcert certificates are not also in the 'User' (user defined) section. From your android device, visit https://www.cacert.org. If you do not see a warning about missing or untrusted certificates, all went well.

Note that some browsers might use their own certificate store instead of the Android one, you might need to import certificate files into those browsers as well.

If you are unable to disable the Android PIN/Pattern lock screen after installing the system certificates, you might need to „Clear/delete credentials“ (in Settings → Security) even though you have removed all user certificates.

If you run into problems, compare the md5 sum of the certificate files with the md5 values in this article, check the file permissions of the newly installed files. Make sure no user certificates are

installed (Settings → Security → Clear certificates), and make sure you are using a browser app that uses the android certificate store and does not implement an own certificate store.

In the future, newer versions of openssl might be used on Android, if so, you might need to drop the „_old“-part of the „-subject_hash_old“ openssl parameter.