Добавление собственного сертификата для всего приложения на Android (Network Security Configuration)

Архив для скачивания:

v2certificates_android.zip

Важно! Для корректной работы необходимо использовать исключительно `WebViewClient`, альтернативные методы взаимодействия с веб-интерфейсами не имеют нужного функционала.

Начиная с версии Android 24 (N) в системе исключена возможность доверять сайтам по умолчанию. В связи с этим появилась возможность добавлять собственные сертификаты через конфигурационные файлы.

Более подробно:

1) https://android-developers.googleblog.com/2016/07/changes-to-trusted-certificate.html

2) https://developer.android.com/training/articles/security-config

Пример загрузки страницы в WebView без и с добавлением сертификата

Рассмотрим добавление сертификата на простом приложении с WebView. В качестве тестового домена был выбран URL:

https://3dsec.sberbank.ru/payment/webservices/merchant-ws?wsdl.

Если попытаться загрузить страницу через WebView *без* добавления сертификата, то в методе *onReceivedSslError* класса *WebViewClient* вернется ошибка *SSL_UNTRUSTED (1).*

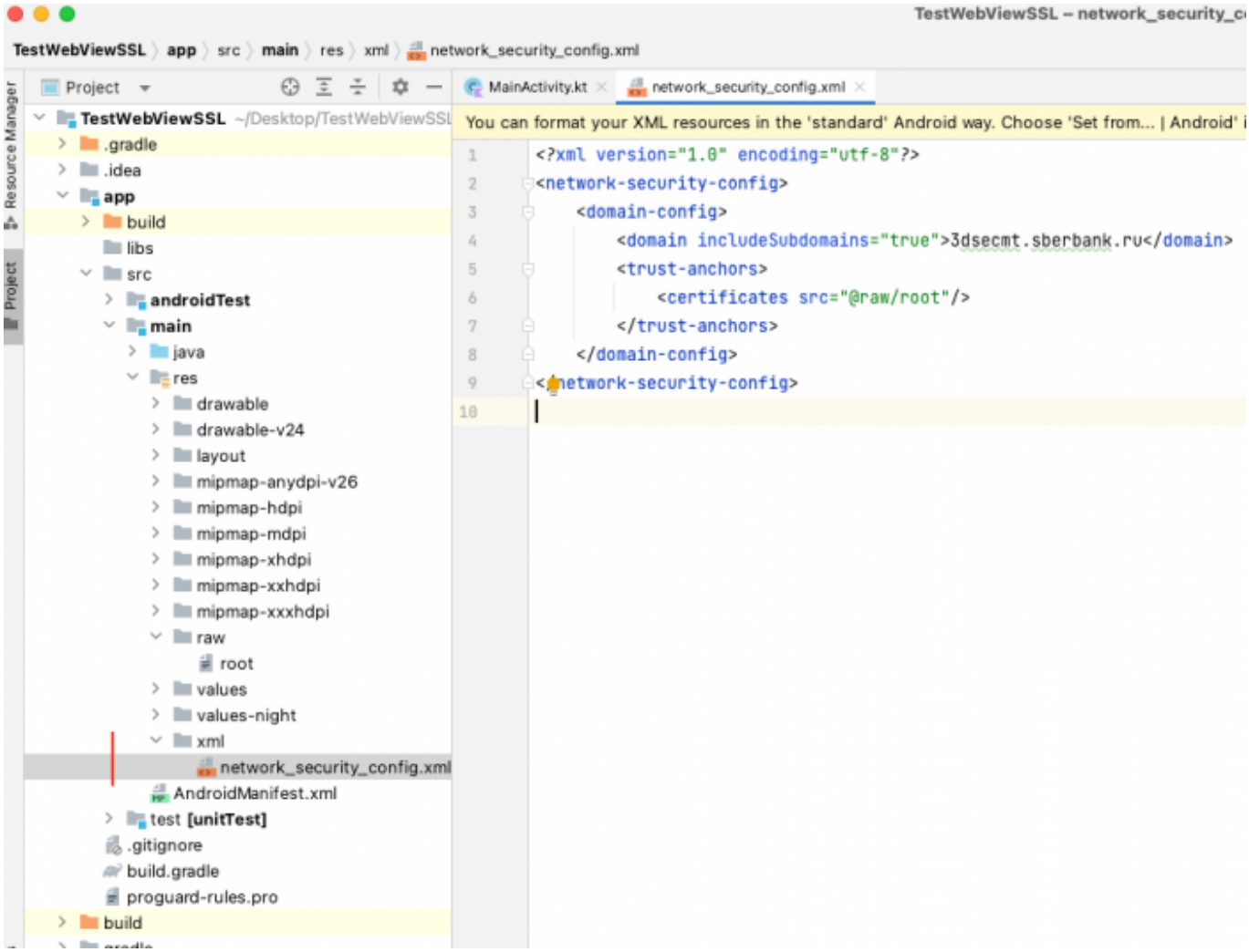
1) Для успешной загрузки страницы, необходимо выполнить следующие шаги для добавления сертификата.

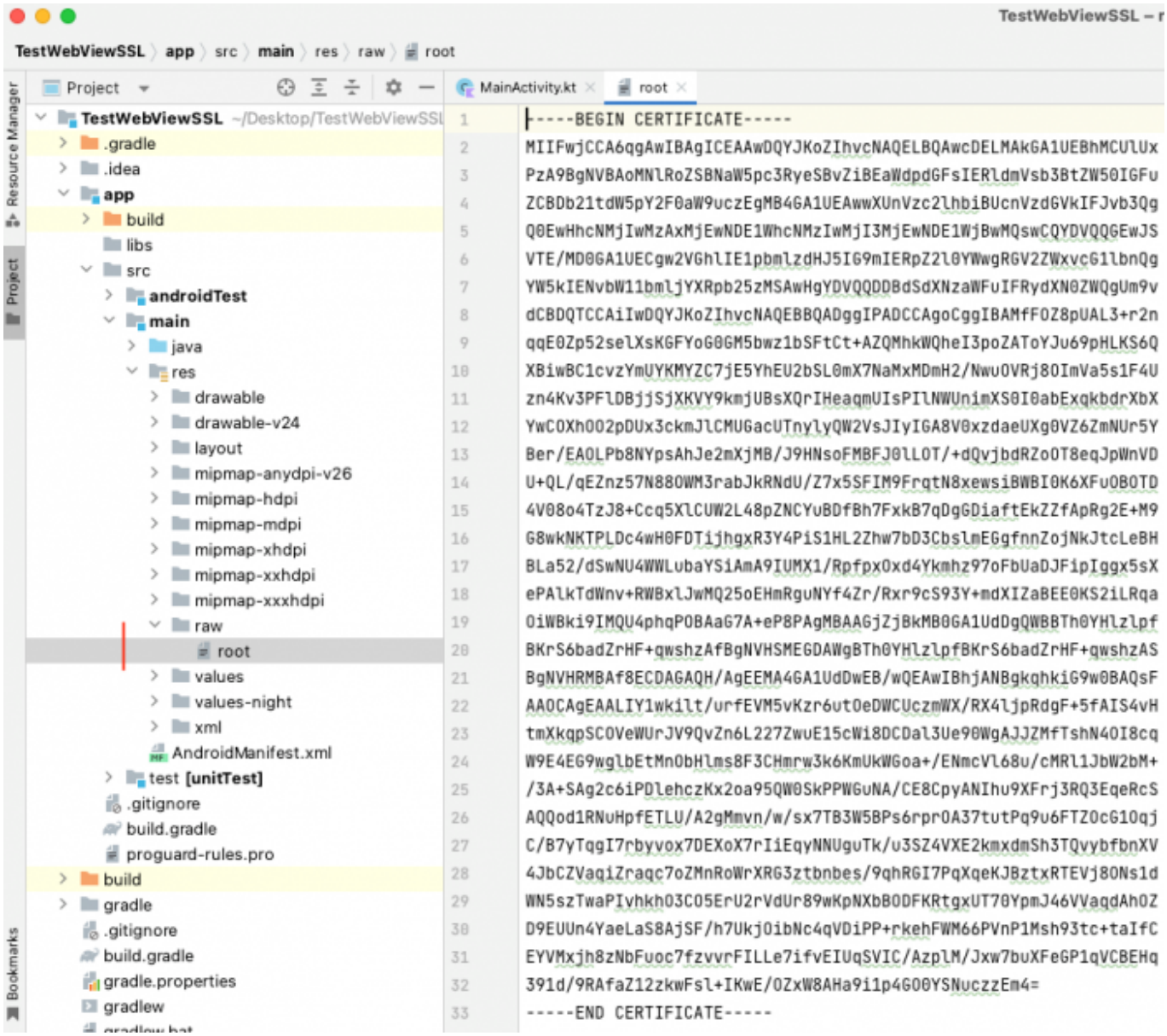
A) В AndroidManifest добавить следующую строчку (доступна с версии API 24,N).

```
<application
        android:networkSecurityConfig="@xml/network_security_config"/>

<uses-permission android:name="android.permission.INTERNET" />

……..
```


TestWebViewSSL – network_security_c
TestWebViewSSL 〉 app 〉 src 〉 main 〉 res 〉 xml 〉 network_security_config.xml
Project
TestWebViewSSL ~/Desktop/TestWebViewSSL
.gradle
.idea
app
build
libs
src
androidTest
main
java
res
drawable
drawable-v24
layout
mipmap-anydpi-v26
mipmap-hdpi
mipmap-mdpi
mipmap-xhdpi
mipmap-xxhdpi
mipmap-xxxhdpi
raw
root
values
values-night
xml
network_security_config.xml
AndroidManifest.xml
test [unitTest]
.gitignore
build.gradle
proguard-rules.pro
build
MainActivity.kt
network_security_config.xml
You can format your XML resources in the 'standard' Android way. Choose 'Set from... | Android'
1 <?xml version="1.0" encoding="utf-8"?>
2 <network-security-config>
3 <domain-config>
4 <domain includeSubdomains="true">3dsecmt.sberbank.ru</domain>
5 <trust-anchors>
6 <certificates src="@raw/root"/>
7 </trust-anchors>
8 </domain-config>
9 </network-security-config>
10



B) Создать каталог raw с файлом сертификата (сертификат должен быть в PEM формате).

C) Далее необходимо создать файл network_security_config.xml в директории xml.

Содержимое файла приведено ниже.

```
<?xml version="1.0" encoding="utf-8"?>
<network-security-config>
    <domain-config>
        <domain includeSubdomains="true">3dsec.sberbank.ru</domain>
        <trust-anchors>
            <certificates src="@raw/root"/>
        </trust-anchors>
    </domain-config>
</network-security-config>
```

Вы можете убрать опцию domain, и тогда сертификат будет работать для любых доменов на

уровне всего приложения.

Если хотим оставить как в примере выше, то убедитесь, что для промышленной сборки, будет добавлен домен: https://securepayments.sberbank.ru.

D) После запуска приложения страница загрузится в WebView без ошибки SSL_UNTRUSTED.

P.S. Корневой сертификат Минцифр из примера был скачан с сайта (Сертификат для Android): https://www.gosuslugi.ru/cr

Альтернативный способ работы с сертификатом МЦ

Также возможен вариант с ручной проверкой сертификатов при первой попытке загрузки страницы в WebView. Для этого необходимо:

1) Подготовить TrustManager.

```
private fun fromInputStream(caInput: InputStream): TrustManagerFactory {
    val cf: CertificateFactory = CertificateFactory.getInstance("X.509")
    val ca = caInput.use { cf.generateCertificate(it) }
    val keyStore: KeyStore =
KeyStore.getInstance(KeyStore.getDefaultType()).apply {
        load(null, null)
        setCertificateEntry("ca", ca)
    }
    val tmfAlgorithm = TrustManagerFactory.getDefaultAlgorithm()
    val tmf = TrustManagerFactory.getInstance(tmfAlgorithm)
    tmf.init(keyStore)
    return  tmf
}
```

2) Реализовать собственный WebViewClient с предопределенным методом onReceivedSslError.

```
override fun onReceivedSslError(view: WebView, handler: SslErrorHandler,
error: SslError) {
    Log.d("WEB_VIEW_EXAMPLE", "onReceivedSslError")
    var passVerify = false
    if (error.primaryError == SslError.SSL_UNTRUSTED) {
        val cert = error.certificate
        val subjectDN = cert.issuedTo.dName
        Log.d("WEB_VIEW_EXAMPLE", "subjectDN: $subjectDN")
        try {
            val f: Field =
cert.javaClass.getDeclaredField("mX509Certificate")
            f.setAccessible(true)
            val x509 = f.get(cert) as X509Certificate
```

```
            val chain = arrayOf(x509)
            for (trustManager in tmf.getTrustManagers()) {
                if (trustManager is X509TrustManager) {
                    val x509TrustManager: X509TrustManager = trustManager as
X509TrustManager
                    try {
                        x509TrustManager.checkServerTrusted(chain,
"generic")
                        passVerify = true
                        break
                    } catch (e: Exception) {
                        Log.e("WEB_VIEW_EXAMPLE", "verify trustManager
failed", e)
                        passVerify = false
                    }
                }
            }
            Log.d("WEB_VIEW_EXAMPLE", "passVerify: $passVerify")
        } catch (e: Exception) {
            Log.e("WEB_VIEW_EXAMPLE", "verify cert fail", e)
        }
    }
    if (passVerify == true) handler.proceed() else handler.cancel()
}
```

Пример можно посмотреть во вложении.