

S4- Sécurité IT et Confiance Numérique

TP : Pentesting d'AndroGoat (Application Android Vulnérable)

Objectif :

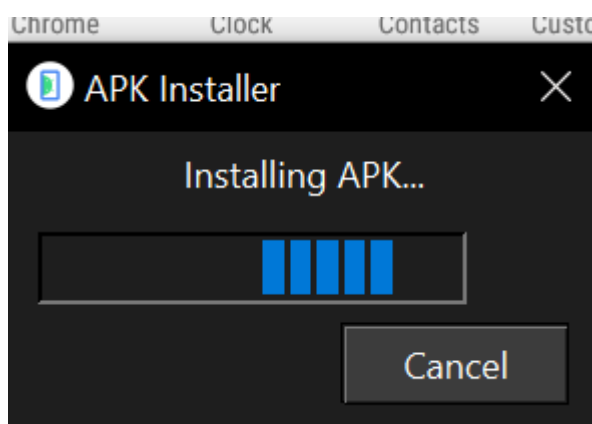
Identifier et exploiter des vulnérabilités dans AndroGoat en utilisant des outils courants de pentesting mobile.

Prérequis :

- Un appareil Android (émulateur ou physique, rooté/débuggable de préférence).
- ADB (Android Debug Bridge).
- Un décompilateur (Jadx-GUI recommandé).
- Un proxy d'interception (Burp Suite Community).
- Application cible : Choisir une application vulnérable (AndroGoat).

1. Configuration du Lab

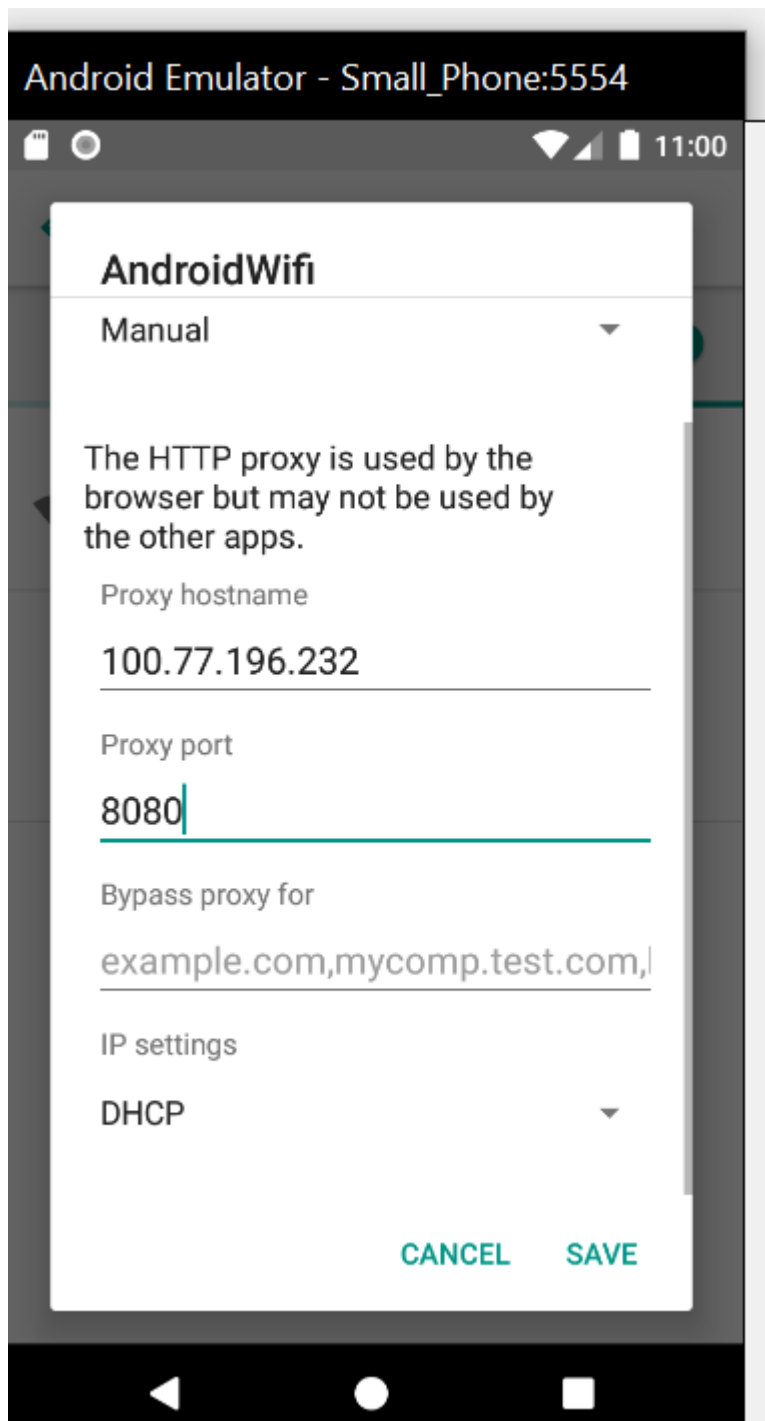
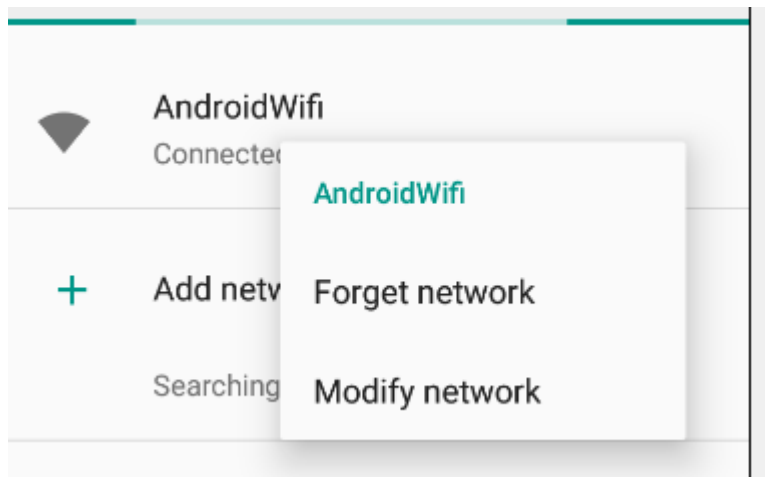
- Étapes :
 - Téléchargez AndroGoat depuis GitHub.
 - Installez l'APK sur un émulateur (Android Studio).



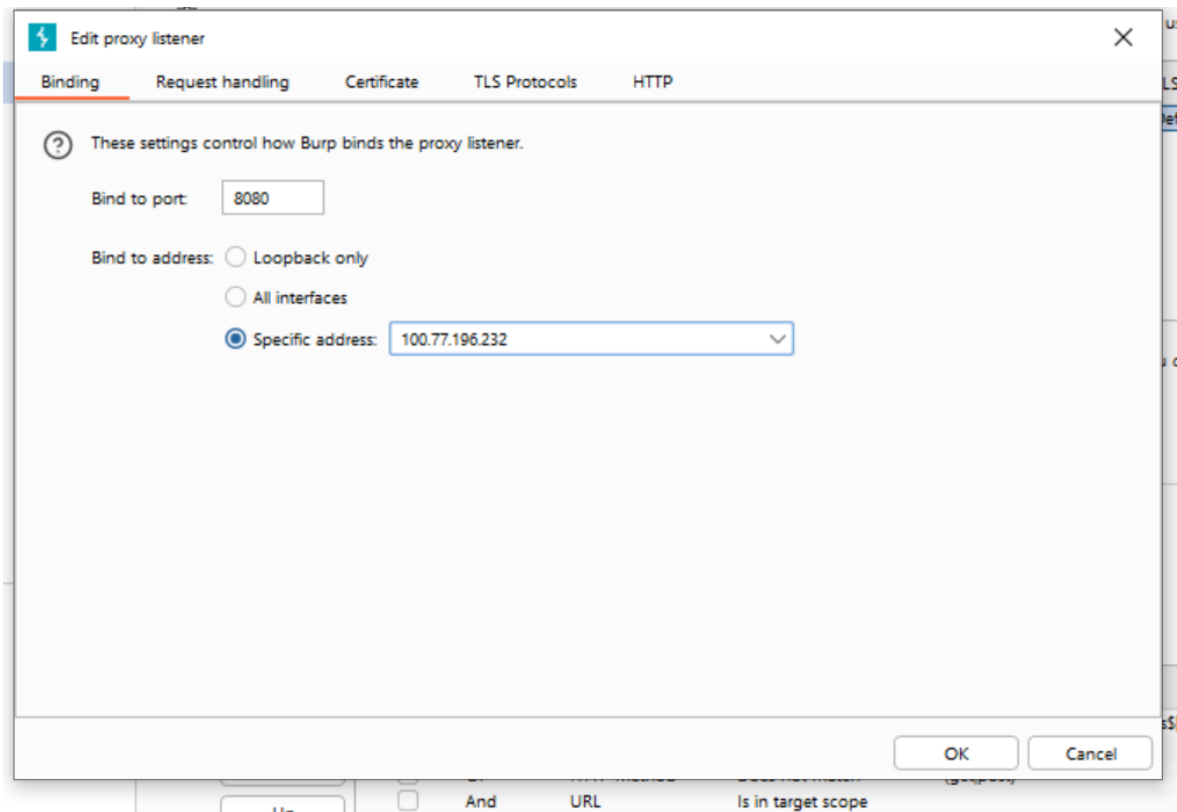
- Configurez ADB (Android Debug Bridge) pour interagir avec l'appareil :
 - `adb devices` # Vérifier la connexion
 - `adb install AndroGoat.apk`

2. Interception du Trafic Réseau

- Configuration d'un Proxy



- Définir l'adresse IP de votre machine comme proxy (ex: 192.168.18.231:8080).



- Dans Burp Suite, configurez un **Proxy Listener** correspondant.
 - Ouvrez AndroGoat et cliquez sur "HTTP" : le trafic sera intercepté dans Burp Suite.



Time	Type	Direction	Method	URL
11:03:37 29 Ap...	HTTP	→ Request	GET	http://demo.testfire.net/

Request		
Pretty	Raw	Hex
1	GET / HTTP/1.1	
2	Host: demo.testfire.net	
3	Connection: keep-alive	
4	Accept-Encoding: gzip, deflate, br	
5	User-Agent: okhttp/3.0.0	
6		
7		

- Pour HTTPS, exportez le certificat Burp (format .cer) et installez-le sur le mobile
 - Accédez aux paramètres Wi-Fi > Certificats > Installez le fichier.
 - Interceptez du trafic HTTPS.



You can export your certificate and key for use in other tools, or in another installation of Burp. You can import a certificate and key to use in this installation of Burp. Note that you can also export the current certificate by visiting <http://burpsuite/cert> in your browser.

Export

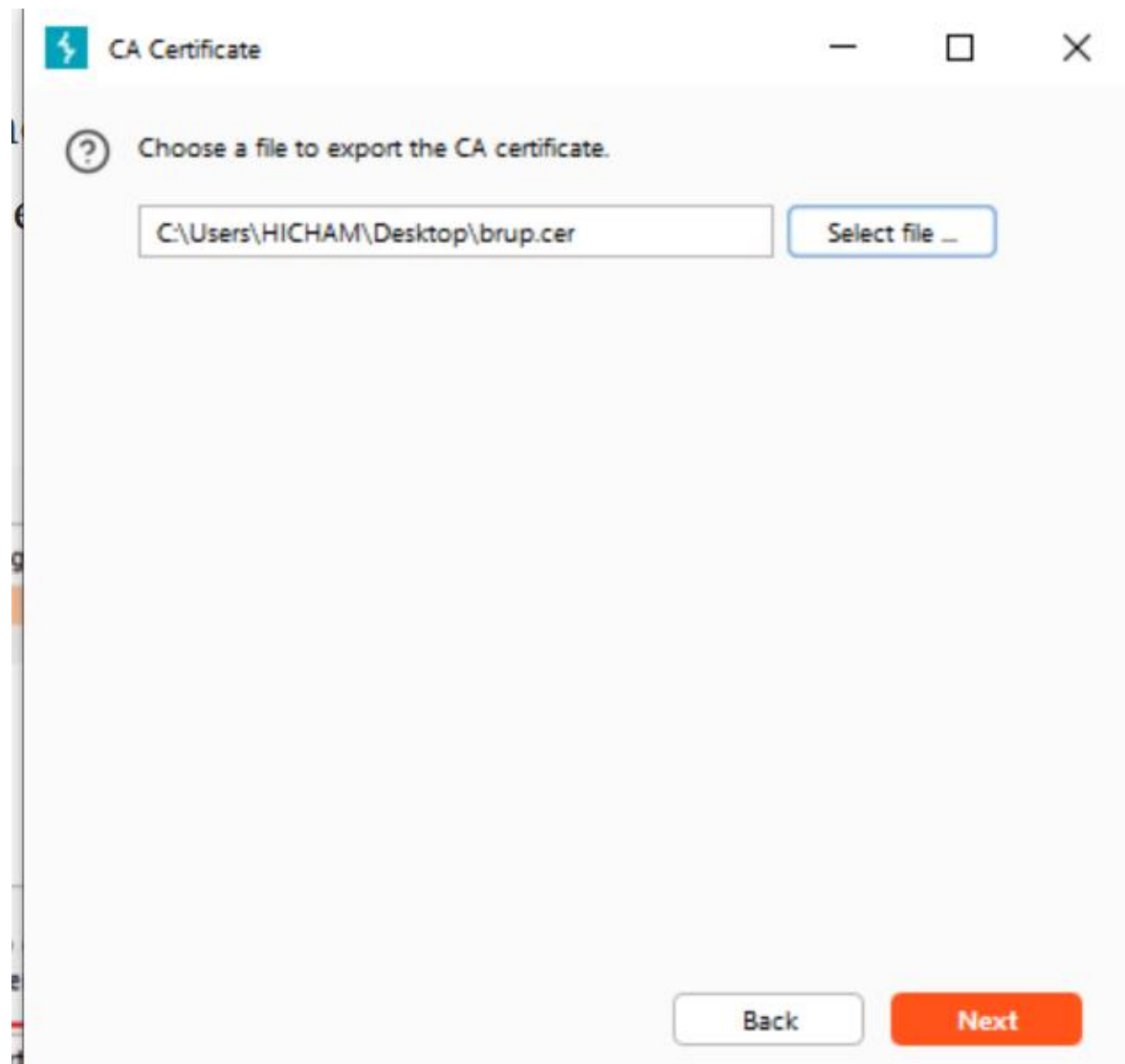
- ☒ Certificate in DER format
- ☐ Private key in DER format
- ☐ Certificate and private key in PKCS#12 keystore

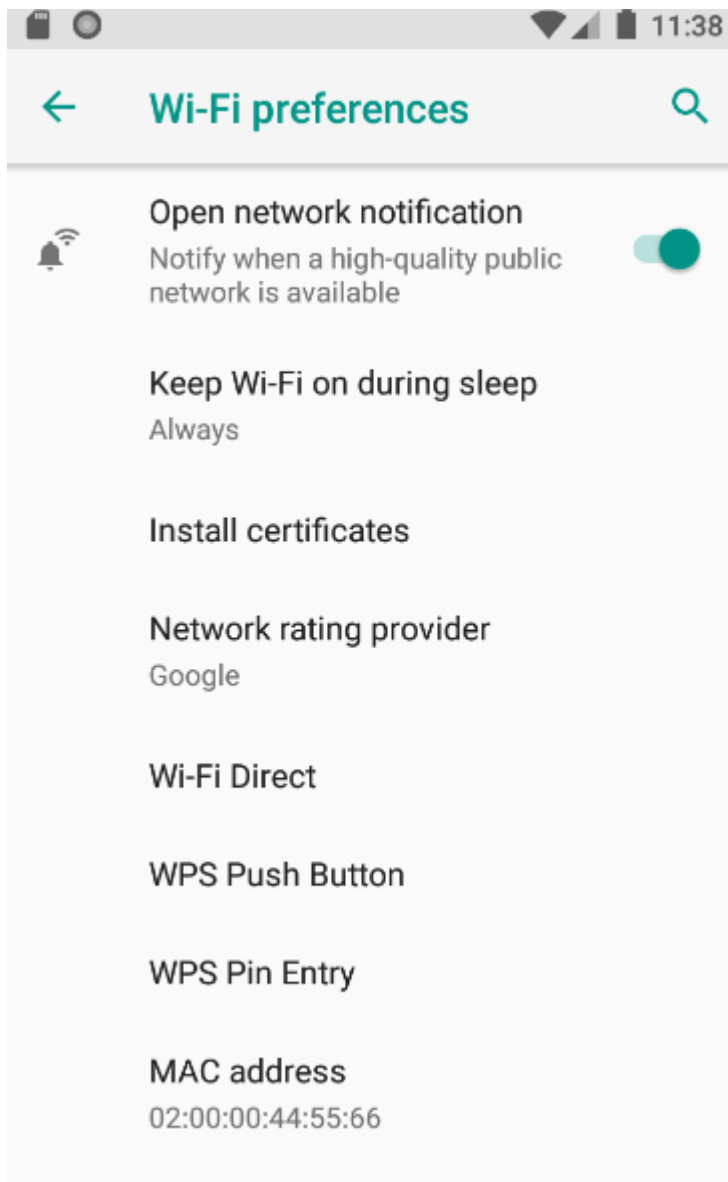
Import

- ☐ Certificate and private key in DER format
- ☐ Certificate and private key from PKCS#12 keystore

Cancel

Next



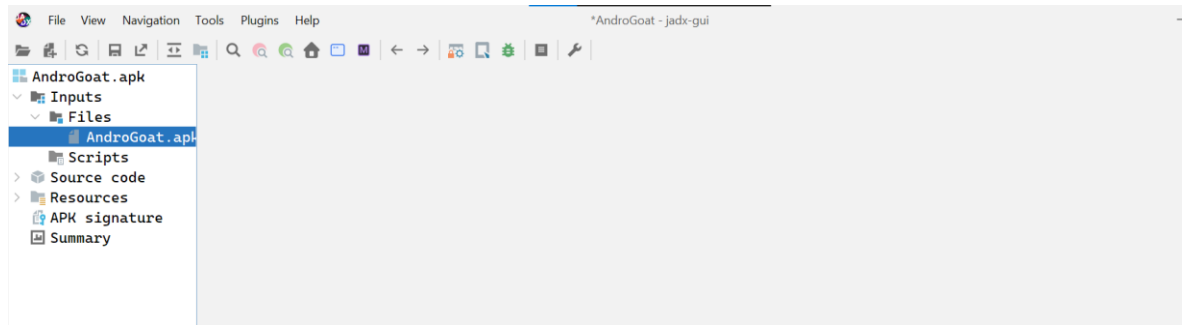


1	http://demo.testfire.net	GET	/	200	9629	HTML	Aktoro Mutual	65.61.137.117	JSESSIONID=680..	11:03:37 29 ..	8080	204
2	http://demo.testfire.net	GET	/	200	9629	HTML	Aktoro Mutual	65.61.137.117	JSESSIONID=OCA..	11:05:23 29 ..	8080	206
3	http://demo.testfire.net	GET	/	200	9629	HTML	Aktoro Mutual	65.61.137.117	JSESSIONID=708..	11:13:45 29 ..	8080	424
4	http://demo.testfire.net	GET	/	200	9629	HTML	Aktoro Mutual	65.61.137.117	JSESSIONID=63C..	11:13:46 29 ..	8080	206
5	http://demo.testfire.net	GET	/	200	9629	HTML	Aktoro Mutual	65.61.137.117	JSESSIONID=043..	11:13:46 29 ..	8080	214
6	http://demo.testfire.net	GET	/	200	9629	HTML	Aktoro Mutual	65.61.137.117	JSESSIONID=464..	11:13:47 29 ..	8080	191
7	http://demo.testfire.net	GET	/	200	9629	HTML	Aktoro Mutual	65.61.137.117	JSESSIONID=4AB..	11:13:47 29 ..	8080	205
8	http://demo.testfire.net	GET	/	200	9629	HTML	Aktoro Mutual	65.61.137.117	JSESSIONID=F43..	11:13:47 29 ..	8080	205
9	http://demo.testfire.net	GET	/	200	9629	HTML	Aktoro Mutual	65.61.137.117	JSESSIONID=55F..	11:13:47 29 ..	8080	204
10	http://demo.testfire.net	GET	/	200	9629	HTML	Aktoro Mutual	65.61.137.117	JSESSIONID=D28..	11:13:48 29 ..	8080	205
11	http://demo.testfire.net	GET	/	200	9629	HTML	Aktoro Mutual	65.61.137.117	JSESSIONID=ED3..	11:13:48 29 ..	8080	198
12	http://demo.testfire.net	GET	/	200	9629	HTML	Aktoro Mutual	65.61.137.117	JSESSIONID=7E9..	11:13:48 29 ..	8080	204
13	http://demo.testfire.net	GET	/	200	9629	HTML	Aktoro Mutual	65.61.137.117	JSESSIONID=571..	11:22:40 29 ..	8080	204
16	http://wajgblivica	HEAD	/				unknown host			11:23:03 29 ..	8080	
17	http://ydzpzhoppu	HEAD	/				unknown host			11:23:03 29 ..	8080	
18	http://heajpela	HEAD	/				unknown host			11:23:03 29 ..	8080	
19	http://clients2.google.com	POST	/service/update2?cup2key=614266..	200	1404	XML		142.250.201.78		11:28:56 29 ..	8080	102
20	http://brup	GET	/					unknown host		11:37:04 29 ..	8080	
22	https://owasp.org	GET	/					172.67.10.39		11:41:21 29 ..	8080	
23	https://owasp.org	GET	/					172.67.10.39		11:41:29 29 ..	8080	
24	https://owasp.org	GET	/					172.67.10.39		11:41:38 29 ..	8080	
25	https://owasp.org	GET	/					172.67.10.39		11:41:41 29 ..	8080	

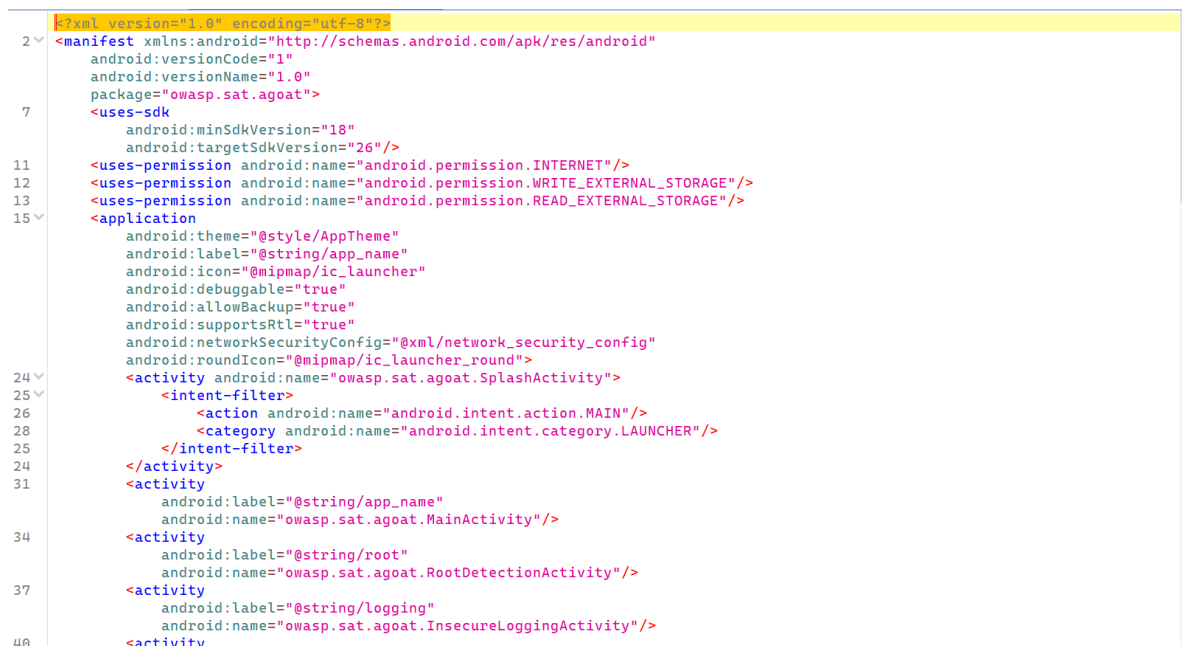
3. Analyse Statique

- Objectif : Comprendre la structure de l'application et identifier des points d'intérêt sans l'exécuter.
- Outil : JADX-GUI. □ Comment faire :

- Ouvrez l'APK d'AndroGoat avec JADX-GUI.



- Examinez le fichier AndroidManifest.xml pour comprendre les permissions demandées, les activités exportées, les services, les broadcast receivers et les content providers. Portez une attention particulière aux intent-filters pour les deeplinks.



comprendre les permissions demandées

```
package="owasp.sat.agoat">
<uses-sdk
    android:minSdkVersion="18"
    android:targetSdkVersion="26"/>
<uses-permission android:name="android.permission.INTERNET"/>
<uses-permission android:name="android.permission.WRITE_EXTERNAL_STORAGE"/>
<uses-permission android:name="android.permission.READ_EXTERNAL_STORAGE"/>
<application
```

INTERNET

WRITE/READ_EXTERNAL_STORAGE

Services


```
<service
    android:name="owasp.sat.agoat.DownloadInvoiceService"
    android:enabled="true"
    android:exported="true"/>
```

- Naviguez dans le code source décompilé (Java). Recherchez des mots-clés suspects (password, key, token, secret, TODO, etc.).

```
Node
android.support.v4.view.accessibility.AccessibilityInfo this.mInfo.setPassword(password);
android.support.v4.view.accessibility.AccessibilityInfo builder.append("password: ");
android.support.v4.view.accessibility.AccessibilityInfo builder.append(isPassword());
android.support.v4.view.accessibility.AccessibilityInfo public boolean isPassword() {
android.support.v4.view.accessibility.AccessibilityInfo return this.mRecord.isPassword();
android.support.v4.view.accessibility.AccessibilityInfo public void setPassword(boolean isPassword) {
android.support.v4.view.accessibility.AccessibilityInfo this.mRecord.setPassword(isPassword);
okhttp3.Credentials.basic(String, String) String public static String basic(String userName, String password) {
okhttp3.Credentials.basic(String, String) String return basic(userName, password, Charset.forName("ISO-8859-1"));
okhttp3.Credentials.basic(String, String, Charset) public static String basic(String userName, String password, Charset charset) {
okhttp3.Credentials.basic(String, String, Charset) String usernameAndPassword = userName + ":" + password;
okhttp3.Credentials.basic(String, String, Charset) byte[] bytes = usernameAndPassword.getBytes(charset);
owasp.sat.agoat.InsecureLoggingActivity.onCreate() final EditText password = (EditText) findViewById(R.id.password);
owasp.sat.agoat.InsecureLoggingActivity.onCreate() sb.append(" and password ");
owasp.sat.agoat.InsecureLoggingActivity.onCreate() EditText password2 = password;
owasp.sat.agoat.InsecureLoggingActivity.onCreate() Intrinsics.checkExpressionValueIsNotNull(password2, "password");
owasp.sat.agoat.InsecureLoggingActivity.onCreate() sb.append((Object) password2.getText());
owasp.sat.agoat.InsecureLoggingActivity.onCreate() sb2.append(" and password ");
owasp.sat.agoat.InsecureLoggingActivity.onCreate() EditText password3 = password;
owasp.sat.agoat.InsecureLoggingActivity.onCreate() Intrinsics.checkExpressionValueIsNotNull(password3, "password");
owasp.sat.agoat.InsecureLoggingActivity.onCreate() sb2.append((Object) password3.getText());
owasp.sat.agoat.InsecureStorageSDCardActivity final EditText password = (EditText) findViewById(R.id.password);
owasp.sat.agoat.InsecureStorageSDCardActivity sb.append(" Password -");
```

```
Node
android.arch.core.internal.SafeIterableMap.get while (currentNode != null && !currentNode.mKey.equals(k)) {
android.arch.core.internal.SafeIterableMap.put protected Entry<K, V> put(K key, V v) {
android.arch.core.internal.SafeIterableMap.put Entry<K, V> newEntry = new Entry<>(key, v);
android.arch.core.internal.SafeIterableMap.put public V putIfAbsent(K key, V v) {
android.arch.core.internal.SafeIterableMap.put Entry<K, V> entry = get(key);
android.arch.core.internal.SafeIterableMap.put put(key, v);
android.arch.core.internal.SafeIterableMap.remove public V remove(K key) {
android.arch.core.internal.SafeIterableMap.remove Entry<K, V> toRemove = get(key);
android.arch.core.internal.SafeIterableMap.remove for (SupportRemove<K, V> iter : this.mIterators.keySet()) {
android.arch.core.internal.SafeIterableMap.remove final K mKey;
android.arch.core.internal.SafeIterableMap.remove Entry<K key, V value> {
android.arch.core.internal.SafeIterableMap.remove this.mKey = key;
android.arch.core.internal.SafeIterableMap.remove return this.mKey.equals(entry.mKey) && this.mValue.equals(entry.mValue);
android.arch.core.internal.SafeIterableMap.remove public K getKey() {
android.arch.core.internal.SafeIterableMap.remove return this.mKey;
android.arch.core.internal.SafeIterableMap.remove return this.mKey + " = " + this.mValue;
android.support.graphics.drawable.AndroidR... public static final int STYLEABLE_KEYFRAME_FRACTION = 3;
android.support.graphics.drawable.AndroidR... public static final int STYLEABLE_KEYFRAME_INTERPOLATOR = 1;
android.support.graphics.drawable.AndroidR... public static final int STYLEABLE_KEYFRAME_VALUE = 0;
android.support.graphics.drawable.AndroidR... public static final int STYLEABLE_KEYFRAME_VALUE_TYPE = 2;
android.support.graphics.drawable.AndroidR... public static final int[] STYLEABLE_KEYFRAME = {android.R.attr.value, android.R.attr.interpolator, android.R.attr.keyframeValue};
android.support.v4.accessibilityservice.AccessibilityService public static final int CAPABILITY_CAN_FILTER_KEY_EVENTS = 8;
android.support.v4.accessibilityservice.AccessibilityService public static final int FLAG_REQUEST_FILTER_KEY_EVENTS = 32;
android.support.v4.accessibilityservice.AccessibilityService return "CAPABILITY_CAN_FILTER_KEY_EVENTS";
android.support.v4.accessibilityservice.AccessibilityService return "FLAG_REQUEST_FILTER_KEY_EVENTS";
android.support.v4.app.BundleCompat.getBinder public static IBinder getBinder(Bundle bundle, String key) {
android.support.v4.app.BundleCompat.getBinder return bundle.getBinder(key);
```

```
Module
android.support.v4.app.Fragment.isVisible() boolean return (!isAdded() || isHidden() || (view = this.mView) == null || view.getWindowToken() == null || this.m
android.support.v4.app.INotificationSideChannel _data.writeInterfaceToken(Stub.DESRIPTOR);
android.support.v4.app.INotificationSideChannel _data.writeInterfaceToken(Stub.DESRIPTOR);
android.support.v4.app.ListFragment.setListAdapter() _data.writeInterfaceToken(Stub.DESRIPTOR);
android.support.v4.app.NotificationCompatSideChannel.setListShown(boolean, boolean) setListShown(true, getView().getWindowToken() != null);
android.support.v4.app.NotificationCompatSideChannel.long idToken = clearCallingIdentity();
android.support.v4.app.NotificationCompatSideChannel.restoreCallingIdentity(idToken);
android.support.v4.app.NotificationCompatSideChannel.long idToken = clearCallingIdentity();
android.support.v4.app.NotificationCompatSideChannel.restoreCallingIdentity(idToken);
android.support.v4.app.NotificationCompatSideChannel.long idToken = clearCallingIdentity();
android.support.v4.app.NotificationCompatSideChannel.restoreCallingIdentity(idToken);
android.support.v4.media.MediaBrowserCompatApi public static Object getSessionToken(Object browserObj) {
android.support.v4.media.MediaBrowserCompatApi return ((MediaBrowser) browserObj).getSessionToken();
android.support.v4.media.MediaBrowserProtocol public static final String DATA_CALLBACK_TOKEN = "data_callback_token";
android.support.v4.media.MediaBrowserProtocol public static final String DATA_MEDIA_SESSION_TOKEN = "data_media_session_token";
android.support.v4.media.MediaBrowserServiceCompat public static void setSessionToken(Object serviceObj, Object token) {
android.support.v4.media.MediaBrowserServiceCompat ((MediaBrowserService) serviceObj).setSessionToken((MediaSession.Token) token);
android.support.v4.media.session.MediaController public static Object fromToken(Context context, Object sessionToken) {
android.support.v4.media.session.MediaController return new MediaController(context, (MediaSession.Token) sessionToken);
android.support.v4.media.session.MediaController public static Object getSessionToken(Object controllerObj) {
android.support.v4.media.session.MediaController return ((MediaController) controllerObj).getSessionToken();
android.support.v4.media.session.MediaSession public static Parcelable getSessionToken(Object sessionObj) {
android.support.v4.media.session.MediaSession return ((MediaSession) sessionObj).getSessionToken();
android.support.v4.media.session.MediaSession public static Object verifyToken(Object token) {
android.support.v4.media.session.MediaSession if (token instanceof MediaSession.Token) {
android.support.v4.media.session.MediaSession return token;
```

```

Kotlin.NumbersKt__NumbersJVMKt.fromBits(Double, return Double.longBitsToDouble(bits);
Kotlin.StandardKt__StandardKt @Metadata(bv = {1, 0, 3}, d1 = {"\u0000\n\u0000\n\u0002\u0010\u0001\n\u0000\n\u0002\u0010\u0000\n\u0000\n"},
Kotlin.StandardKt__StandardKt.TODO() Void private static final Void TODO() {
Kotlin.StandardKt__StandardKt.TODO(String) Vo private static final Void TODO(String reason) {
Kotlin.UByte @Metadata(bv = {1, 0, 3}, d1 = {"\u0000\n\u0002\u0010\u0002\n\u0002\u0010\u000f\n\u0000\n\u0002\u0010\u0000\n"},
Kotlin.UByte /* renamed from: toDouble-impl, reason: not valid java name */
Kotlin.UByte.m56toDoubleimpl(byte) double private static final double m56toDoubleimpl(byte $this) {
Kotlin.UInt @Metadata(bv = {1, 0, 3}, d1 = {"\u0000\n\u0002\u0010\u0002\n\u0002\u0010\u000f\n\u0000\n\u0002\u0010\u0000\n"},
Kotlin.UInt /* renamed from: toDouble-impl, reason: not valid java name */
Kotlin.UInt.m125toDoubleimpl(int) double private static final double m125toDoubleimpl(int $this) {
Kotlin.UInt.m125toDoubleimpl(int) double return UnsignedKt.unsignedToDouble($this);
Kotlin.UInt.m126toFloatimpl(int) float return (float) UnsignedKt.unsignedToDouble($this);
Kotlin.ULong @Metadata(bv = {1, 0, 3}, d1 = {"\u0000\n\u0002\u0010\u0002\n\u0002\u0010\u000f\n\u0000\n\u0002\u0010\u0000\n"},
Kotlin.ULong /* renamed from: toDouble-impl, reason: not valid java name */
Kotlin.ULong.m194toDoubleimpl(long) double private static final double m194toDoubleimpl(long $this) {
Kotlin.ULong.m194toDoubleimpl(long) double return UnsignedKt.unsignedToDouble($this);
Kotlin.ULong.m195toFloatimpl(long) float return (float) UnsignedKt.unsignedToDouble($this);
Kotlin.UShort @Metadata(bv = {1, 0, 3}, d1 = {"\u0000\n\u0002\u0010\u0002\n\u0002\u0010\u000f\n\u0000\n\u0002\u0010\u0000\n"},
Kotlin.UShort /* renamed from: toDouble-impl, reason: not valid java name */
Kotlin.UShort.m289toDoubleimpl(short) double private static final double m289toDoubleimpl(short $this) {
Kotlin.UnsignedKt @Metadata(bv = {1, 0, 3}, d1 = {"\u0000\n\u0000\n\u0002\u0010\u0000\n\u0002\u0010\u0000\n\u0002\u0002\n"},
Kotlin.UnsignedKt.doubleToInt(double) int if (Double.isNaN(v) || v <= unsignedToDouble(0)) {
Kotlin.UnsignedKt.doubleToULong(double) long if (v >= unsignedToDouble(-1)) {
Kotlin.UnsignedKt.doubleToULong(double) long if (!Double.isNaN(v) && v > unsignedToDouble(0L)) {
Kotlin.UnsignedKt.doubleToULong(double) long if (v >= unsignedToDouble(-1L)) {
Kotlin.UnsignedKt.unsignedToDouble(int) double public static final double unsignedToDouble(int v) {
Kotlin.UnsignedKt.unsignedToDouble(long) double public static final double unsignedToDouble(long v) {

```

Tutoriel à suivre : Pentesting Mobile : Guide pas à pas de l'évaluation AndroGoat

Liens : <https://infosecwriteups.com/mobile-pentesting-androgoat-assessment-walkthrough-1a63a7edc677>