APK Analysis Report
Android TV

The AndroidTV application was decompiled and analyzed. While the application contains **standard R8/ProGuard obfuscation**, the code structure is mainly not obfuscated. The application logic can be comprehended. **No advanced encryption or anti-reverse engineering measures** were noted.

Basic Information

- **Package Name**: com.example.tvpasswordprotection
- **App Name**: Android TV
- **Target SDK**: 34 (Android 14)
- **Min SDK**: 26 (Android 8.0)
- **Application Type**: Android TV Password Protection System
- **Application Class:** com.example.tvpasswordprotection.utill.App
- **Main Activities:**
    - MainActivity2 (Primary launcher)
    - MainActivity
    - Settings Activity
- **Version**: 1.0

Key Permissions

```xml
<!-- Device Administration -->
<uses-permission android:name="android.permission.BIND_DEVICE_ADMIN" />

<!-- Network & Internet -->
<uses-permission android:name="android.permission.INTERNET" />
<uses-permission android:name="android.permission.ACCESS_NETWORK_STATE" />

<!-- VPN Services -->
<uses-permission android:name="android.permission.BIND_VPN_SERVICE" />

<!-- Accessibility -->
<uses-permission android:name="android.permission.BIND_ACCESSIBILITY_SERVICE" />

<!-- System Level -->
<uses-permission android:name="android.permission.FOREGROUND_SERVICE" />
<uses-permission android:name="android.permission.SYSTEM_ALERT_WINDOW" />
```

Obfuscation Analysis

1. Standard Obfuscation Present

The application uses **standard R8/ProGuard obfuscation** with the following characteristics:

Obfuscated Elements:

- **Lambda expressions**: MainActivity$onCreate$1, MainActivity$onCreate$2
- **Inner classes**: NotificationRepoImp$Companion
- **Synthetic methods**: Generated getter/setter methods

Example of Obfuscated Code:

File: \AndroidTV_src\smali\com\example\tvpasswordprotection\MainActivity$onCreate$1.smali

```
.class public final
Lcom/example/tvpasswordprotection/MainActivity$onCreate$1;
.super Lkotlin/jvm/internal/Lambda;
.source "SourceFile"

# interfaces
.implements Lkotlin/jvm/functions/Function1;

# annotations
.annotation system Ldalvik/annotation/Signature;
    value = {
        "Lkotlin/jvm/internal/Lambda;",
        "Lkotlin/jvm/functions/Function1<",
        "Landroid/view/View;",
        "Lkotlin/Unit;",
        ">;"
    }
.end annotation
```

2. Readable Code Structure

Despite obfuscation, **main application logic remains readable**:

Main Activity Implementation:

File: AndroidTV_src\smali\com\example\tvpasswordprotection\MainActivity.smali

```
.class public final Lcom/example/tvpasswordprotection/MainActivity;
.super Landroidx/appcompat/app/AppCompatActivity;
.source "SourceFile"

# Key methods remain identifiable:
.method public onCreate(Landroid/os/Bundle;)V
.method public onDestroy()V
.method public onKeyDown(ILandroid/view/KeyEvent;)Z
```

3. Password Fragment:

Clear Password Logic Handling.

File: \AndroidTV_src\smali\com\example\tvpasswordprotection\PasswordFragment.smali

```
.class public final Lcom/example/tvpasswordprotection/PasswordFragment;
.super Landroidx/fragment/app/Fragment;
.source "SourceFile"

# Clear password handling logic
.method public final validatePassword()V
.method public final setupPasswordInput()V
```

Key Application Components

1. Device Administration

File: AndroidTV_src\smali\com\example\tvpasswordprotection\device_owner\DeviceOwnerReceiver.smali

```
.class public final
Lcom/example/tvpasswordprotection/device_owner/DeviceOwnerReceiver;
.super Landroid/app/admin/DeviceAdminReceiver;
.source "SourceFile"
 .method public
onEnabled(Landroid/content/Context;Landroid/content/Intent;)
```

In Android Manifest

```
<receiver
android:name="com.example.tvpasswordprotection.device_owner.DeviceOwnerReceiver"
          android:permission="android.permission.BIND_DEVICE_ADMIN">
```

2. VPN Service Implementation

File: AndroidTV_src\smali\com\example\tvpasswordprotection\vpn\DummyVpnService.smali

```
.class public final Lcom/example/tvpasswordprotection/vpn/DummyVpnService;
.super Landroid/net/VpnService;
.source "SourceFile"

.method public onCreate()V
.method public onStartCommand(Landroid/content/Intent;II)I
```

- Implements a dummy VPN service
- Uses fake HTTP requests to Google (generate_204) to keep VPN alive
- VPN configuration: 10.0.0.2/32 with 0.0.0.0/0 routing
- Purpose appears to be traffic interception/monitoring

3. Network Communication

File: AndroidTV_src\smali\com\example\tvpasswordprotection\apis\NotificationRepoImp.smali

Clear API endpoint visible:

```
# Direct field
.field public static final BASE_URL:Ljava/lang/String;

.method public final sendNotification(Ljava/lang/String;)V
.method public final checkServerStatus()V



.line 16

    const-string v0, "https://f577-103-137-24-80.ngrok-free.app/
```

4. Accessibility Service

File: AndroidTV_src\smali\com\example\tvpasswordprotection\accessibility\AccessibilityHelper.smali

```
.class public final Lcom/example/tvpasswordprotection/accessibility/AccessibilityHelper;
.super Landroid/accessibilityservice/AccessibilityService;

.method public onAccessibilityEvent(Landroid/view/accessibility/AccessibilityEvent;)V
.method public onServiceConnected()V
```

In Android Manifest.xml

```
<service android:name="com.example.tvpasswordprotection.MyForegroundService"
         android:permission="android.permission.BIND_ACCESSIBILITY_SERVICE">
```

5. String Literals

String literals are **completely unobfuscated:**

File: DummyVpnService.smali

- "DummyVPN"
- "VPN Running"
- "https://www.google.com/generate_204"
- "10.0.0.2"
- "0.0.0.0"
- "vpn_channel_id"

File: Contstant.smali

- "prefs"
- "isFirst"
- "myPassword"
- "mySwitch"
- "isWifiEnable"

Library Dependencies Analysis

Standard Libraries Detected:

1. **OkHttp3** - HTTP client library
2. **Retrofit2** - REST API client
3. **Koin** - Dependency injection framework
4. **AndroidX** - Android support libraries
5. **Kotlin Coroutines** - Asynchronous programming

Library Code Status:

All third-party libraries are in standard, unmodified form - no custom obfuscation or encryption was applied to external dependencies.

Security Assessment

1. No Advanced Obfuscation Detected

- **No string encryption** found
- **No dynamic code loading** detected
- **No native code obfuscation** present
- **No anti-debugging measures** implemented
- **No custom packing/encryption** found

2. Readable Resource Strings

File: \AndroidTV_src\res\values\strings.xml

```
<string name="app_name">TV Password Protection</string>
<string name="password_prompt">Enter Password:</string>
<string name="invalid_password">Invalid password. Please try again.</string>
<string name="device_admin_label">TV Protection Admin</string>
<string name="vpn_service_label">TV Protection VPN</string>
```

3. Hardcoded Values Visible

Several sensitive configurations are **directly readable**:

- API endpoints: "https://f577-103-137-24-80.ngrok-free.app/
- Service names and labels
- Password validation logic flows

Findings

1. Application Logic Flow

The password protection system implements:

- Device admin privileges for system-level control
- VPN service for network monitoring
- Accessibility service for UI interaction blocking
- Foreground service for persistent operation

2. Network Communication

- **Base URL**: "https://f577-103-137-24-80.ngrok-free.app/
- **Protocol**: Standard HTTP/HTTPS using OkHttp3
- **Data Format**: Likely JSON (Retrofit2 suggests REST API)

3. Security Mechanisms

- Password-based access control
- Device administrator enforcement
- System overlay blocking
- Network traffic monitoring via VPN

Final Result:

The AndroidTV.apk is NOT heavily protected against reverse engineering. The application uses only standard R8/ProGuard obfuscation, which presents minimal barriers to analysis.

The application's security relies primarily on runtime privilege enforcement rather than code protection, making it fully accessible for security analysis and reverse engineering s.

Analysis Date: July 4, 2025
Analysis Method: Static APK analysis using APKTool