SECURITY ANALYSIS OF ANDROID AUTOMOTIVE
Mert D. Pesé and Kang G. Shin, University of Michigan
Josiah Bruner, Georgia Institute of Technology
Amy Chu, Harman International
Agenda

Introduction
Related Work
Threat Model and Background
Security Analysis
Recommendations
Next Generation of IVIs

Android Auto: The First Great In-Car Infotainment System

Google Unveils Android Automotive OS on the 2020 Polestar 2 EV
By Ryan Whitwam on May 3, 2019 at 2:15 pm 4 Comments

Source: https://www.wired.com/2015/05/android-auto-first-great-car-infotainment-system/
Source: https://www.extremetech.com/mobile/290792-google-unveils-android-automotive-os-on-the-2020-polestar-2
Android Auto vs Android Automotive

Android Auto

• Runs outside vehicles (on phone)
• Phone connection required, since mirroring
  • Cannot use data from IVN
  • Only restricted to media and messaging apps

Android Automotive

• Runs inside vehicles (on IVI)
• No phone connection required
  • Can use data from IVN
  • Richer 3rd party apps possible

+ Restricted Permissions
+ Restricted Attack Surfaces
- Phone Integration

+ No Phone
- More Attack Surfaces
- Access to IVN data
→ Data Injection & Privacy

Source: https://www.funzen.net/2019/11/20/how-android-auto-works-everything-you-need-to-know/

Agenda

Introduction
Related Work
Threat Model and Background
Security Analysis
Recommendations
Related Work

Android Auto
- Static analysis of infotainment apps in Google Play Store
- Vulnerabilities limited to operational damage, but also driver safety (distraction)
- Study found 60% of all apps have some sort of vulnerability – 25% of all apps have JavaScript vulnerabilities

Android Automotive
- Focus on third-party app analysis
- Developed tool for vehicle-specific code analysis
- PoC attacks for driver disturbance, availability, privacy
Agenda

- Introduction
- Related Work
- Threat Model and Background
- Security Analysis
- Recommendations
Classification of Attacks

Attack Landscape is changing...

First-Generation Attacks
(~2010-2015)
Using physical interfaces

Second-Generation Attacks
(~2015-2020)
Using wireless interfaces
(e.g., IVI and TCU)

Third-Generation Attacks
(~2020-?)
Using app eco-system on IVIs

Scalability

Risk / Damage Potential
Classification of Attacks

... so is the risk.

Infrastructure Vulnerabilities
- Network
- OS
- Runtime Environment
- Hypervisor
- Backend

Can be patched

(Programming) Framework Design Vulnerabilities
- APIs
- Permission Model
- Unauthorized Access to IVN

Far-reaching impact, significant disruption once adapted system

Early analysis and disclosure to Google is vital!
Permission Model

Four levels of protection level
• Normal: No explicit consent needed
• Dangerous: Explicit user consent required
• Signature: Cryptographically signed with platform certificate
• signature|privileged: Cryptographically signed or pre-installed

Third-party applications only have access to normal and dangerous permissions 😊
47 permissions defined in android.car.permission as of October 2019

<table>
<thead>
<tr>
<th>Permission Name</th>
<th>Protection Level</th>
<th>Permission Name</th>
<th>Protection Level</th>
</tr>
</thead>
<tbody>
<tr>
<td>READ_CAR_DISPLAY_UNITS</td>
<td>Normal</td>
<td>CAR_MOCK_VEHICLE_HAL</td>
<td>signature</td>
</tr>
<tr>
<td>CONTROL_CAR_DISPLAY_UNITS</td>
<td>Normal</td>
<td>READ_CAR_STEERING</td>
<td>signature</td>
</tr>
<tr>
<td>CAR_ENERGY_PORTS</td>
<td>Normal</td>
<td>CAR_IDENTIFICATION</td>
<td>signature</td>
</tr>
<tr>
<td>CAR_INFO</td>
<td>Normal</td>
<td>CAR_MILEAGE</td>
<td>signature</td>
</tr>
<tr>
<td>CAR_EXTERNAL_ENVIRONMENT</td>
<td>Normal</td>
<td>CAR_TIRES</td>
<td>signature</td>
</tr>
<tr>
<td>CAR_POWERTRAIN</td>
<td>Normal</td>
<td>CARENGINE_DETAILED</td>
<td>signature</td>
</tr>
<tr>
<td>CAR_SPEED</td>
<td>Dangerous</td>
<td>CAR_DYNAMICS_STATE</td>
<td>signature</td>
</tr>
<tr>
<td>CAR_ENERGY</td>
<td>Dangerous</td>
<td>CAR_VENDOR_EXTENSION</td>
<td>signature</td>
</tr>
<tr>
<td>BIND VMS_CLIENT</td>
<td>Signature</td>
<td>CAR_PROJECTION</td>
<td>signature</td>
</tr>
<tr>
<td>BIND_PROJECTION_SERVICE</td>
<td>Signature</td>
<td>ACCESS_CAR_PROJECTION_STATUS</td>
<td>signature</td>
</tr>
<tr>
<td>BIND_INSTRUMENT_CLUSTER_RENDERER_SERVICE</td>
<td>Signature</td>
<td>CONTROL_CAR_SEATS</td>
<td>signature</td>
</tr>
<tr>
<td>BIND_CAR_INPUT_SERVICE</td>
<td>Signature</td>
<td>CONTROL_CAR_MIRRORS</td>
<td>signature</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CONTROL_CAR_WINDOWS</td>
<td>signature</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CONTROL_CAR_DOORS</td>
<td>signature</td>
</tr>
<tr>
<td></td>
<td></td>
<td>CONTROL_CAR_CLIMATE</td>
<td>signature</td>
</tr>
</tbody>
</table>
Vehicle Properties

Implemented by VHAL

Vendor-extendable Android module to abstract vehicle data for SDK, APK

Mapping properties to CAN signals provided by DBCs
Agenda

Introduction
Related Work
Threat Model and Background
Security Analysis
Recommendations
Create PoC attacks based on severity classification of EVITA

<table>
<thead>
<tr>
<th>Security threat severity class</th>
<th>Safety</th>
<th>Privacy</th>
<th>Financial</th>
<th>Operational</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>No injuries</td>
<td>No unauthorized access to data</td>
<td>No financial loss</td>
<td>No impact on operational performance</td>
</tr>
<tr>
<td>1</td>
<td>Light or moderate injuries</td>
<td>Anonymous data only (no specific driver of vehicle data)</td>
<td>Low-level loss ($\approx \infty$ 10)</td>
<td>Impact not discernible to driver</td>
</tr>
<tr>
<td>2</td>
<td>Severe injuries (survival probable); light/moderate injuries for multiple vehicles</td>
<td>Identification of vehicle or driver; anonymous data for multiple vehicles</td>
<td>Moderate loss ($\approx \infty$ 100); low losses for multiple vehicles</td>
<td>Driver aware of performance degradation; indiscernible impacts for multiple vehicles</td>
</tr>
<tr>
<td>3</td>
<td>Life threatening (survival uncertain) or fatal injuries; severe injuries for multiple vehicles</td>
<td>Driver or vehicle tracking; identification of driver or vehicle for multiple vehicles</td>
<td>Heavy loss ($\approx \infty$ 1000); moderate losses for multiple vehicles</td>
<td>Significant impact on performance; noticeable impact for multiple vehicles</td>
</tr>
<tr>
<td>4</td>
<td>Life threatening or fatal injuries for multiple vehicles</td>
<td>Driver or vehicle tracking for multiple vehicles</td>
<td>Heavy losses for multiple vehicles</td>
<td>Significant impact for multiple vehicles</td>
</tr>
</tbody>
</table>

SAE INTERNATIONAL                      2020-01-1295
Attack #1: Privacy

Goal: Malicious 3rd party app obtains privacy-sensitive driver information

Speed has *dangerous* permission
- Explicit user consent necessary

Gear position and RPM have *normal* permission
- Can be read by any app without user consent

**Speed = f(gear, RPM)**

Dangerous permission is circumvented
- More examples possible
- Physical signals have certain relationships with each other...

Source: [http://homepages.bw.edu/~katchins/csc131common/a_papers/student2/gearmath.htm](http://homepages.bw.edu/~katchins/csc131common/a_papers/student2/gearmath.htm)
Goal: Malicious 3rd party app breaks instrument cluster

CONTROL_CAR_DISPLAY_UNITS has *normal* permission

- Display units for distance, fuel, tire pressure, EV battery, fuel consumption can be modified

Examples: Switch from min. to max. fuel level, force TPMS light to come on etc.

- Bound by 1 Hz frequency (1 change per second)

Financial damage: Needle will break eventually

Operational damage: Driver realizes something is wrong with tires and brings car to dealership/tire shop

### Attack #3: Safety

**Goal:** Malicious 3rd party app accelerates the vehicle instead of displaying value on instrument cluster

**Not all CAN signals mapped to vehicle properties**
- Acceleration/Gas pedal does not need to be read/written

**Option #1: Reverse engineering of the IVI FW**
- DBCs and mapping table are stored on IVI
- Change mapping
- Reflash

**Option #2: Access via ADB shell**

Agenda

Introduction
Related Work
Threat Model and Background
Security Analysis
Recommendations
Fine-grained permission model

- Problem: Multiple properties summarized in one permission
- Assign unique permission for property
- Quantify privacy risk of each property, assign protection levels accordingly

Further standardization from Google

- Problem: Vendors given too much free space for implementation design
- Google should define security recommendations and standardize more modules
- Example: DBC mapping without physically storing DBC file, use lookup table in Trusted Execution Environment (TEE)
Recommendations

Separation of domains in IVN architecture
- Problem: IVI might control other (safety-critical) ECUs
- Implement access control, e.g., by firewall, in gateway

Protection against runtime attacks
- Problem: Android still susceptible to Return-Oriented Programming (ROP) attacks, can lead to buffer overflows
- Vendor-specific C/C++ code (device drivers, etc.) most vulnerable

Restrict ADB shell access (USB and WiFi!)
- Disable USB debugging by default in production
- Never allow default user to run as root
THANK YOU

Mert D. Pesé
University of Michigan – Ann Arbor
2260 Hayward Street, Ann Arbor, MI 48109-2121, U.S.A

mpese@umich.edu