Published in Mobile Dev Blog by Bamboo Apps

Bamboo Apps Follow
Jul 25, 2018 · 8 min read · Listen

Automotive Cybersecurity Best Practices (part 1)

How to ensure the highest possible degree of security in the era of the intelligent connected car.

Disclaimer

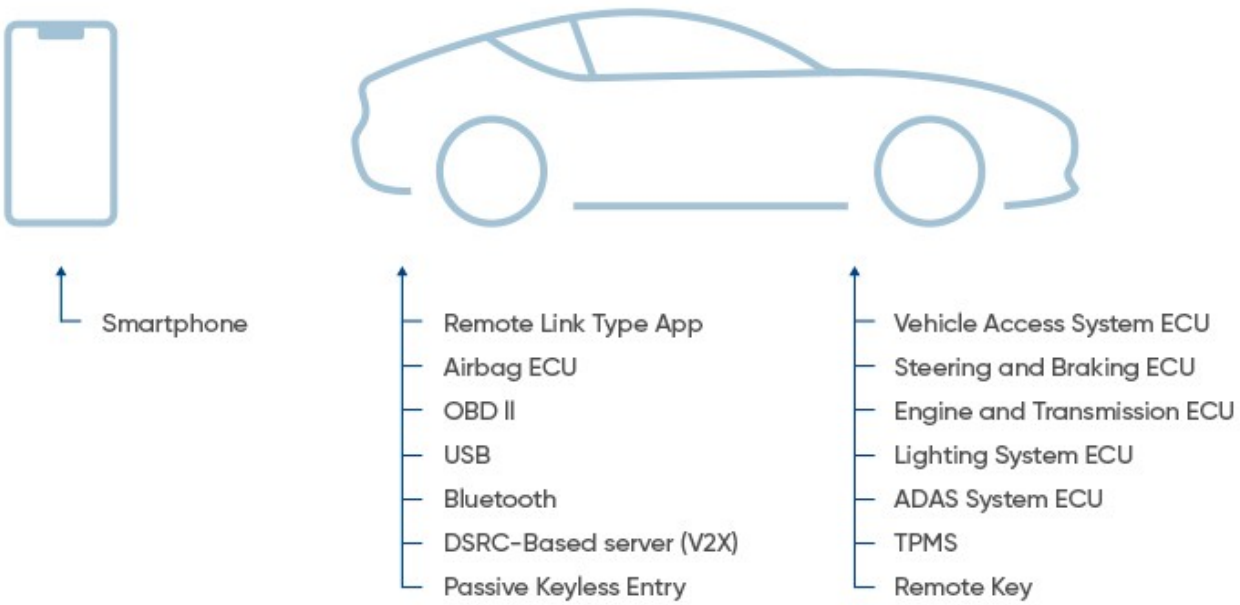
You may read the whole story on Bamboo Apps' website. For this, please download our e-book.

payment, and entertainment. Increased automation, vehicle-to-vehicle, and vehicle-to-infrastructure communications add cybersecurity and data privacy as cornerstones for consumer confidence and vehicle safety.

Working with OEMs and Tier 1 suppliers around the world, we developed an engineering approach that delivers a high degree of back-end and front-end security to connected car software. In this post, we are sharing this experience with the automotive community.

Automotive Cybersecurity Challenge

One of the central challenges in vehicle cybersecurity is that the various electrical components in a car (known as electronic control units, or ECUs) are connected via an internal network. Thus, if hackers manage to gain access to a vulnerable, peripheral ECU — for instance, a car's Bluetooth or infotainment system — from there they may be able to take control of critical ECUs like its brakes or engine.



Main Hackable Attack Surface of the Vehicle

- **Hardware security.** Hardware security services build on top of hardware security and provide fast cryptographic performance, immutable device identification, message authentication, and execution isolation.
- **Software security**. Software security services enhance security capabilities on top of the hardware with network enforcement, whitelists/blacklists, anomaly detection, cryptographic services, biometrics, secure over-the-air updates, and upgrade capabilities, all delivered over the life of the car.
- **Network security**. With in-vehicle networks carrying a mix of operational and personally identifiable information — such as location, navigation history, call history, microphone recordings — protecting messages and data is critical for operational security and privacy.
- **Cloud security.** While embedded vehicle security is essential, some additional security services require real-time intelligence and updates, so the systems need to be able to connect to cloud-based security services in order to detect and correct threats before they get to the car.

In this post, we will go through core aspects of **automotive software security** in terms of back-end and front-end development.

Back-end Security Techniques

As the volume of data to be processed and distributed in connected cars increases, software developers must deploy in-depth approaches allowing to protect back-end systems. At Bamboo Apps for each of the problems and goals of backend security design, we use numerous components and techniques.




Secure Rest API Communication

HTTPS

REST (or REpresentational State Transfer) is an architectural style which is evolved into the HTTP/1.1 and URI specs and has been proven to be well-suited for developing distributed hypermedia applications. While REST is more widely applicable, it is most commonly used within the context of communicating with services via HTTP.

Secure REST services must only provide HTTPS endpoints. This protects authentication credentials in transit, for example, passwords, API keys or JSON Web Tokens. Based on our experience, it also allows clients to authenticate the service and guarantee the integrity of the transmitted data.

> *When working with automotive companies on connected car projects we are using mutually authenticated client-side certificates to provide additional protection for highly privileged web services.*

Architecture

Software architects on our back-end team carefully think out what architecture works best for a specific client's solution. We start from choosing the appropriate method to protect data when it is being transmitted. For this, we opt for Virtual Private Networks (VPN) or a Transport Layer Security model (TLS model). This technology protects web application data from unauthorized disclosure and modification when it is transmitted between clients and the server. The model can be easily adapted to client's business needs. For example, a VPN connection may serve as the best solution to provide a mutual access to a shared server over a variety of protocols. Conversely, an Internet-facing enterprise web application would likely be better served by a TLS model.

Transport layer protection is necessary for back-end connections and any other connection where sensitive data is exchanged or where user identity is established. **TLS is mainly a defense against man-in-the-middle attacks.** TLS protocols are also effective in securing communications with external entities including telematics service providers, consumer smart devices and, in future, other vehicles and ITS infrastructure.

The server validation component of TLS provides authentication of the server to the client. If configured to require client-side certificates, TLS can also play a role in client

controls to prevent tampering with any portion of the encrypted data. In addition, controls are also built-in to prevent a captured stream of TLS data from being replayed at a later time.

It should be noted that TLS provides the above guarantees to data during transmission. TLS does not offer any of these security benefits to data that is at rest. Therefore appropriate security controls must be added to protect data while at rest within the application or within data stores.

The basic requirements for using TLS are: access to a Public Key Infrastructure (PKI) in order to obtain certificates, access to a directory or an Online Certificate Status Protocol (OCSP) responder in order to check certificate revocation status, and agreement/ability to support a minimum configuration of protocol versions and protocol options for each version.

Server Certificate

Establishing a certificate authority and managing certificates for servers allows each entity within the infrastructure to validate the other members' identity and encrypt their traffic. To prevent man-in-the-middle attacks Bamboo Apps abides by the following Server Certificate Rules:

- We use strong keys & protect them;
- We use a certificate that supports required domain names;
- We use fully qualified names in certificates;
- We do not use wildcard certificates;
- We do not use RFC 1918 Addresses in Certificates;
- We use an appropriate certification authority for the application's user base;
- We always provide all needed certificates;
- We are aware of and have a plan for the SHA-1 deprecation plan.




Rule — Only Support Strong Protocols

SSL/TLS is a collection of protocols. The best practice for transport layer protection is to only provide support for the TLS protocols — TLS 1.0, TLS 1.1 and TLS 1.2. This configuration will provide maximum protection against skilled and determined attackers and is appropriate for applications handling sensitive data or performing critical operations.

Rule — Prefer Ephemeral Key Exchanges

When working with automotive companies on connected car projects we use Ephemeral key exchanges that are based on Diffie-Hellman and use per-session, temporary keys during the initial SSL/TLS handshake. They provide perfect forward secrecy (PFS), which means a compromise of the server's long-term signing key does not compromise the confidentiality of past session. When the server uses an ephemeral key, the server signs the temporary key with its long-term key (the long-term key is the customary key available in its certificate).

Rule — Only to support strong Cryptographic Ciphers

Each protocol (TLSv1.0, TLSv1.1, TLSv1.2, etc) provides cipher suites. The strength of the encryption used within a TLS session is determined by the encryption cipher negotiated between the server and the browser. In order to ensure that only strong cryptographic ciphers are selected the server must be modified to disable the use of weak ciphers and to configure the ciphers in an adequate order. It is recommended to configure the server to only support strong ciphers and to use sufficiently large key sizes. In general, the following are observed by our team when selecting cipher suites:

- Use the very latest recommendations, they may be volatile these days;
- Setup policy to get a whitelist for recommended ciphers;
- Verify a cipher string;
- Inform yourself how to securely configure the settings for your used services or hardware;




When using a shared secret or password offer TLS-PSK (Pre-Shared Key) or TLS-SRP (Secure Remote Password), which are known as Password Authenticated Key Exchange (PAKEs). TLS-PSK and TLS-SRP properly bind the channel, which refers to the cryptographic binding between the outer tunnel and the inner authentication protocol. Basic authentication places the user's password on the wire in the plain text after a server authenticates itself. Basic authentication only provides unilateral authentication. In contrast, both TLS-PSK and TLS-SRP provide mutual authentication, meaning each party proves it knows the password without placing the password on the wire in the plain text. Using a PAKE removes the need to trust an outside party, such as a Certification Authority (CA).

Rule — Only Support Secure Renegotiations

A design weakness in TLS, identified as CVE-2009–3555, allows an attacker to inject a plaintext of his choice into a TLS session of a victim. In the HTTPS context, the attacker might be able to inject his own HTTP requests on behalf of the victim. The issue can be mitigated either by disabling support for TLS renegotiations or by supporting only renegotiations compliant with RFC 5746. All modern browsers have been updated to comply with this RFC.

Rule — Disable Compression

Compression Ratio Info-leak Made Easy (CRIME) is an exploit against the data compression scheme used by the TLS and SPDY protocols. The exploit allows an adversary to recover user authentication cookies from HTTPS. The recovered cookie can be subsequently used for session hijacking attacks.

Rule — Update Crypto Libraries

- Use solely versions of crypto libraries that are still supported;
- Watch out for vulnerabilities and update your crypto libraries on a regular base.

Access Control

Explicitly permit communications and messages only between pre-approved systems




In Bamboo Apps' software development practice, access control is performed by REST services at each API endpoint. Web services in monolithic applications implement this by means of user authentication, authorisation logic and session management. This has several drawbacks for modern architectures which compose multiple microservices following the RESTful style.

- In order to minimize latency and reduce coupling between services, the access control decision should be taken locally by REST endpoints;
- User authentication should be centralized in an Identity Provider (IdP), which issues access tokens.

This post is the first in a series of publications about automotive cybersecurity. Next posts are coming soon, stay in touch!