Published in Mobile Dev Blog by Bamboo Apps

Bamboo Apps Follow
Aug 1, 2018 · 8 min read · Listen

Automotive Cybersecurity Best Practices (part 3)

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

This article is the final part of a series about *Automotive Cybersecurity Best Practices*. If you haven't got the chance to read the previous ones, here are some convenient links: Part 1 and Part 2.

Disclaimer

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

As auto manufacturers increasingly rely on software to evolve the connected and autonomous vehicle landscape, they cannot afford to be complacent when it comes to front-end security. With the approaches of effective back-end vehicle security defined above, the next step is to ensure trustworthy mechanisms and techniques, we implement for front-end security.

Up-to-date Cryptographic Algorithms

Cryptography plays an especially important role in securing the user's data - even more so in a mobile environment, where attackers having physical access to the user's device is a likely scenario. When assessing a mobile app, it’s necessary to make sure that it does not use cryptographic algorithms and protocols that have significant known weaknesses or are otherwise insufficient for modern security requirements.

Algorithms that were considered secure in the past may become insecure over time; therefore, it's still important to periodically check current best practices and adjust configurations accordingly.

For this reason, Bamboo Apps recommends using the following algorithms:

- Confidentiality algorithms: AES-GCM-256 or ChaCha20-Poly1305;
- Integrity algorithms: SHA-256, SHA-384, SHA-512, Blake2;
- Digital signature algorithms: RSA (3072 bits and higher), ECDSA with NIST P-384;
- Key establishment algorithms: RSA (3072 bits and higher), DH (3072 bits or higher), ECDH with NIST P-384.

Strong Key Generation Functions

Cryptographic algorithms (such as symmetric encryption or some MACs) expect a secret input of a given size. For example, AES uses a key of exactly 16 bytes. A native implementation might use the user-supplied password directly as an input key. Bamboo




- If the password is smaller than the key, the full key space isn't used. The remaining space is padded (spaces are sometimes used for padding);
- A user-supplied password will realistically consist mostly of displayable and pronounceable characters. Therefore, only some of the possible 256 ASCII characters are used and entropy is decreased by approximately a factor of four.

With this in mind, when using password derivation functions, we propose to opt for an appropriate iteration count (for example, NIST recommends and an iteration count of at least 10,000 for PBKDF2).

Strong Random Number Generators

Pseudo-random number generators (RNG) produce a stream of pseudo-random numbers — a stream of numbers that appear as if they were randomly generated. The quality of the generated numbers varies with the type of algorithm used. Cryptographically secure RNGs generate random numbers that pass statistical randomness tests and are resilient against prediction attacks.

> *In order to ensure the quality of the generated numbers, our team supports SecureRandom implementation on Android side and SecureRandomBytes implementation on iOS.*

Password Strength

Password strength is a key concern when passwords are used for authentication. The password policy defines requirements to which end users should adhere. A password policy typically specifies password length, password complexity, and password topologies. A "strong" password policy makes manual or automated password cracking difficult or impossible.

In an effort to secure strong password policy, we comply with the following requirements for password length:

- Minimum password length (10 characters) should be enforced;
- Maximum password length should not be too short because it will prevent users from creating passphrases. The typical maximum length is 128 characters




- At least one uppercase character (A-Z);
- At least one lowercase character;
- At least one digit (0–9);
- At least one special character.

Login Throttling

On the development stage of any automotive project, a source code needs to be checked for a throttling procedure: a counter for logins attempted in a short period of time with a given username and a method to prevent login attempts after the maximum number of attempts has been reached. After an authorized login attempt, the error counter should be reset.

When implementing brute force mitigation techniques, Bamboo Apps' has identified four proven practices preventing logging attacks:

- After a few unsuccessful login attempts, targeted accounts we lock targets (temporarily or permanently), and additional login attempts should be rejected.
- We also use a five-minute account lock for temporary account locking.
- To prevent client-side controls from being easily bypassed, we implement additional controls on the server.
- Unauthorized login attempts will tally with respect to the targeted account, not a particular session.

Data Encryption on the Network

Bamboo Apps is aware that one of the core mobile app functions is sending/receiving data over untrusted networks like the Internet. This is why it is essential to rely on HTTP for communication with the backend. HTTPS wraps HTTP in an encrypted connection (the acronym HTTPS originally referred to HTTP over Secure Socket Layer (SSL); SSL is the deprecated predecessor of TLS). TLS allows authentication of the backend service and ensures confidentiality and integrity of the network data.




As an example in this regard we can mention the popular Android networking library OkHttp that uses the following preferred set of cipher suites (available only on Android versions 7.0 and later):

- TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256;
- TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256;
- TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384;
- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384;
- TLS_ECDHE_ECDSA_WITH_CHACHA20_POLY1305_SHA256;
- TLS_ECDHE_RSA_WITH_CHACHA20_POLY1305_SHA256.

Similarly, the iOS ATS (App Transport Security) configuration requires one of the following ciphers:

- TLS_ECDHE_ECDSA_WITH_AES_256_GCM_SHA384;
- TLS_ECDHE_ECDSA_WITH_AES_128_GCM_SHA256;
- TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA384;
- TLS_ECDHE_ECDSA_WITH_AES_256_CBC_SHA;
- TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA256;
- TLS_ECDHE_ECDSA_WITH_AES_128_CBC_SHA;
- TLS_ECDHE_RSA_WITH_AES_256_GCM_SHA384.
- TLS_ECDHE_RSA_WITH_AES_128_GCM_SHA256;
- TLS_ECDHE_RSA_WITH_AES_256_CBC_SHA384;
- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA256;
- TLS_ECDHE_RSA_WITH_AES_128_CBC_SHA.




when an unauthorized party is able to view and modify all traffic passing between the mobile device and the back-end systems.

> *Bamboo Apps uses certificate pinning to verify that a certificate comes from a trusted source and to check whether the endpoint server presents the right certificate.*

On Android side from version 7.0, we use the Network Security Configuration feature, to customize their network security settings in a safe, declarative configuration file without modifying app code.

Network Security Configuration (NSC) feature is also used by our team to perform declarative certificate pinning on specific domains.

Testing for Sensitive Functionality Exposure through IPC Overview

During the process of development of a mobile application, Bamboo Apps applies traditional techniques for IPC, in particular, the use of shared files or network sockets. As mobile application platforms implement their own system functionality for IPC, these mechanisms should be applied as we consider them much more mature than traditional techniques. As shown by our experience, using IPC mechanisms with security in mind can cause the application to leak or expose sensitive data.

The list below shows Android IPC Mechanisms that may expose sensitive data:

- Binders;
- Services;
- Bound;
- Services;
- AIDL;
- Intents;
- Content Providers.

- Dynamic analysis.

Protect to Store Sensitive Data in the Keyboard Cash

In order to simplify keyboard input, Bamboo Apps offers several options, like providing autocorrection or spell checking. Most of the keyboard input is cached by default. The application must ensure that data typed into text fields which contain sensitive information are not cached, it is achieved by disabling the feature programmatically.

Sensitive Data in Backups

iOS and Android offer auto-backup features that create copies of the data on the device. On iOS, backups should be made either through iTunes, or the cloud using the iCloud backup feature. In both cases, the backup includes nearly all data stored on the device, except for some highly sensitive things (like Apple Pay, Google Pay information and Touch ID settings).

Since an application backs up installed apps and their data, an obvious concern is whether sensitive user data stored by the app might unintentionally leak through the backup.

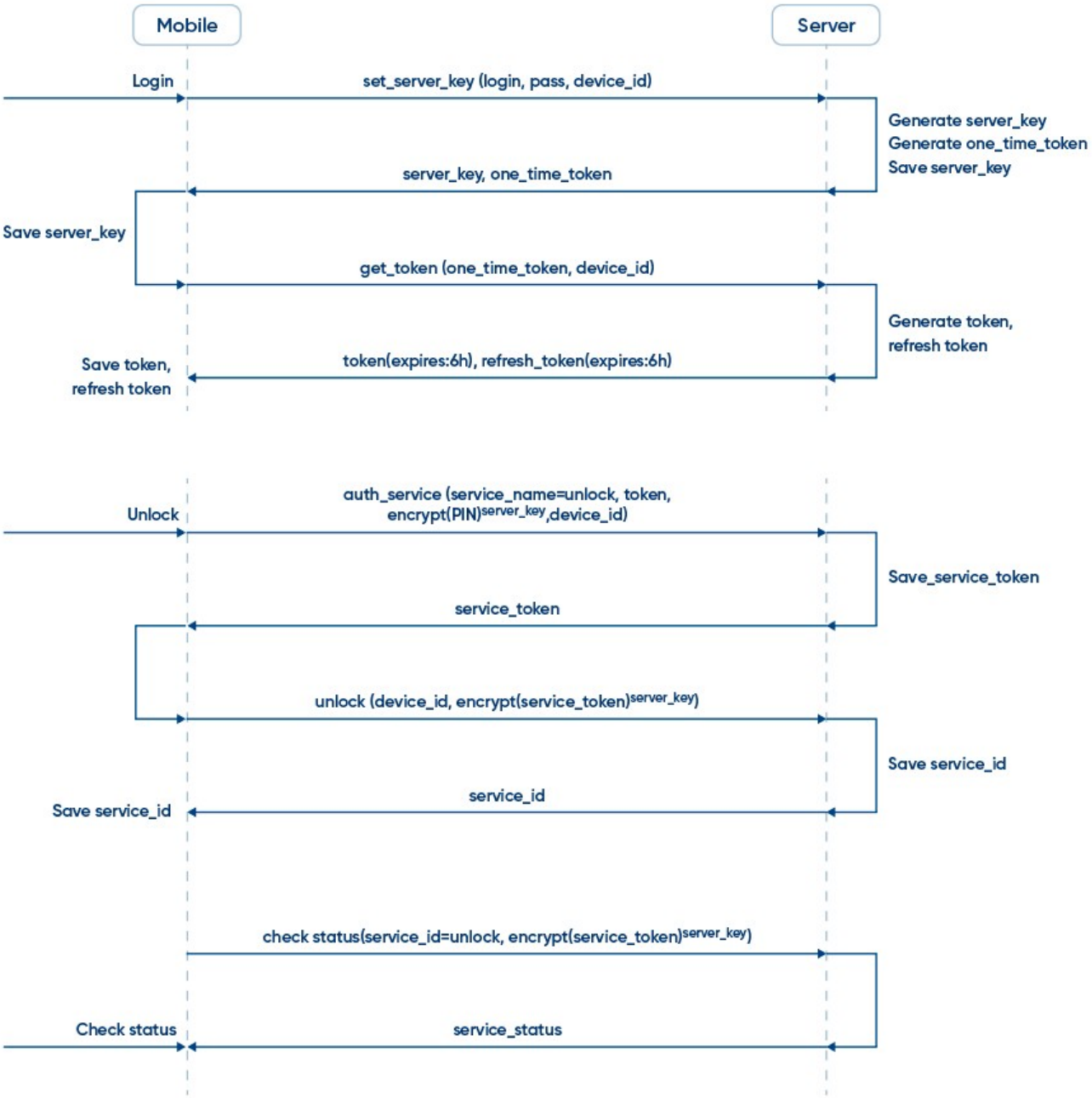
Sensitive Information on Screenshots

Manufacturers want to provide device users an aesthetically pleasing effect when an application is entered or exited, hence they introduced the concept of saving a screenshot when the application goes into the background. This feature could potentially pose a security risk for an application, as the screenshot containing sensitive information (e.g. a screenshot of an email or corporate documents) is written to local storage, where it can be recovered either by a rogue application on a jailbroken device or by someone who steals the device. For this very reason, Bamboo Apps uses a default screenshot that is cached whenever the application enters the background.

Network Communication for Car Unlock Operation

Communication starts after the secure HTTPS connection is established and certificate pinning process is finished. There are two phases of the communication: an initial login



Bamboo Apps' approach for car unlocking operation

Conclusion

Modern vehicles are a point of growing concern among drivers, acting as mobile access points to sensitive personal data and entrusted with the physical security of the passengers within them.

working on standardization and improvement of security measures to mitigate the security risk to a vehicle.

In pursuing this goal, we apply a range of practices that can help improve security at the back-end stage of development. PKI-based authentication mechanisms assist in securing internal communications, whereas transport layer security (TLS) can be effective in securing communications with external entities. The authenticity is verified by using digital certificates and confidentiality is maintained by encrypting all communication.

Providing front-end security requires the array of approaches commonly used in a mobile environment. When assessing a mobile app, up-to-date cryptographic algorithms are employed to secure the user's data. A strong password policy, used for authentication, makes manual or automated password cracking difficult or impossible. Utilizing multiple tools, such as login throttling, certificate pinning, protecting sensitive data and information on screenshots are recommended to limit the extent of the threat and avoid cyber attacks.

About Bamboo Apps

Bamboo Apps is a Design & Software Studio with a proven background in the automotive domain. We develop connected solutions for key players in the automotive industry, mobility, and enterprise security.

We have over 6 years of experience in designing and engineering secure software for the OEMs. We assist our clients with infotainment, HMI, telematics solutions to be implemented in intelligent connected vehicles with the aim of ensuring the highest possible degree of security.