Published in Mobile Dev Blog by Bamboo Apps

Bamboo Apps Follow
Jul 30, 2018 · 4 min read · Listen

Save

Automotive Cybersecurity Best Practices (part 2)

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

This is Part 2 of *Automotive Cybersecurity Best Practices* which continues a series of publications about automotive cybersecurity. In case you just stumbled across this, you should probably start with Part 1.

Disclaimer

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

API Keys

API key is a code passed in by computer programs calling an application programming interface (API) to identify the calling program, its developer, or its user to the Web site. API keys are used to track and control how the API is being used, for example, to prevent malicious use or abuse of the API.

> *Based on our experience, public REST services without access control run the risk of being farmed leading to excessive bills for bandwidth or compute cycles. API keys can be used to mitigate this risk.*

API keys will also reduce the impact of denial-of-service attacks. However, when they are issued to third-party clients, they are relatively easy to compromise. Our main duties when using API keys are the following:

- API keys for every request to the protected endpoint will be required;
- 429 "Too Many Requests" HTTP response code will be returned if requests are coming in too quickly;
- API key will be revoked if the client violates the usage agreement;
- We do not rely exclusively on API keys to protect sensitive, critical or high-value resources.

Secure HTTPS Headers

Our main recommendations for HTTP headers modifications:

- Set the Content-Security-Policy header to help prevent cross-site scripting attacks and other cross-site injections;
- Remove the X-Powered-By header;
- Add Public Key Pinning headers to prevent man-in-the-middle attacks with forged certificates;
- Set Strict-Transport-Security header that enforces secure (HTTP over SSL/TLS) connections to the server;

- Set the X-Frame-Options header to provide clickjacking protection;
- Set X-XSS-Protection to enable the Cross-site scripting (XSS) filter in most recent web browsers.

Safetynet Attestation API

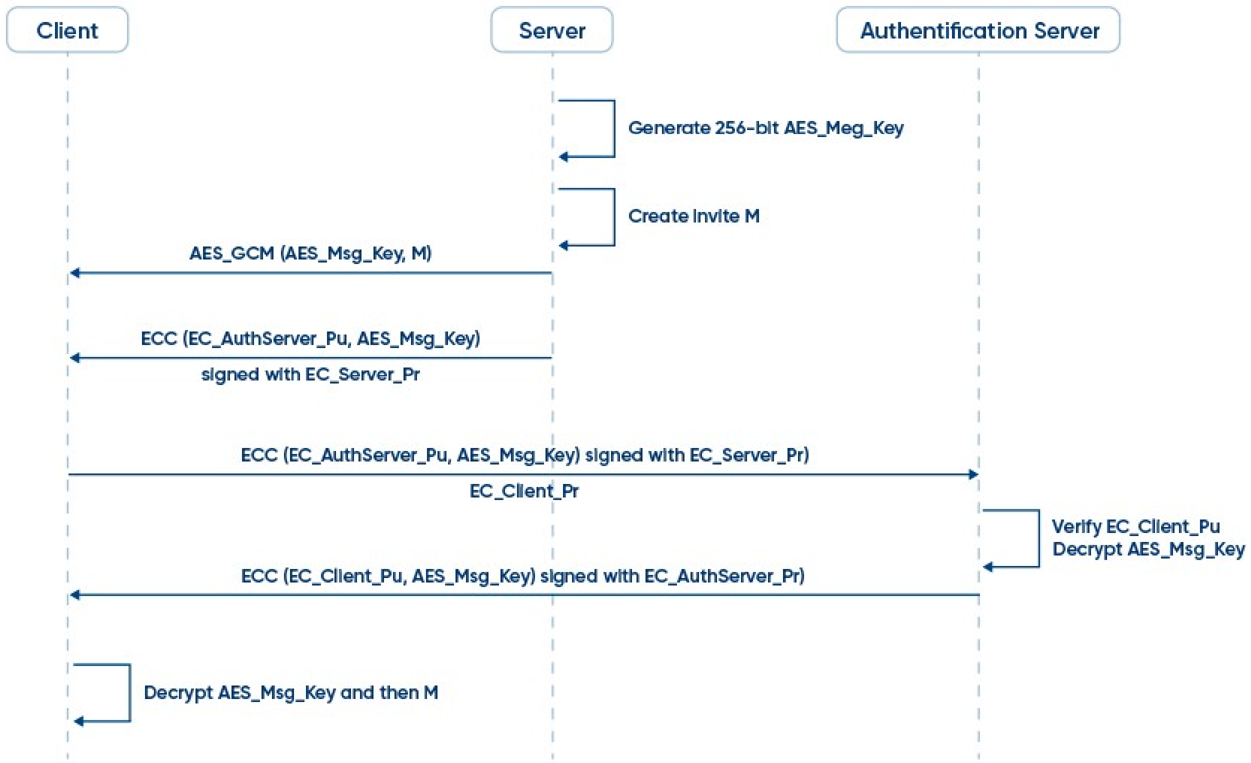
The SafetyNet Attestation API helps assess the security and compatibility of the mobile environment in which an app runs. We use this API to analyze devices that have installed an app.

SafetyNet examines software and hardware information on the device where an app is installed to create a profile of that device. The service then attempts to find this same profile within a list of device models that have passed Android compatibility testing. The API also uses this software and hardware information to help assess the basic integrity of the device, as well as the APK information. This attestation helps to determine whether or not the particular device has been tampered with or otherwise modified. It also provides information about the app that is using this API so that you can assess whether the calling app is legitimate.

This API is to provide us with confidence about the integrity of a device running an app. We use the SafetyNet Attestation API as an additional in-depth defense signal as part of an anti-abuse system, not as the sole anti-abuse signal for your app.

Custom Authentication Scheme

Based on our experience in working with authentication services, we are convinced of the effectiveness of using an additional authentication server. This server provides a complete cost-effective multi-factor authentication, password replacement, and single sign-on authentication solution for traditional desktops, over the phone, in the browser or from any device.


Client
Server
Authentification Server
Generate 256-bit AES_Meg_Key
Create invite M
AES_GCM (AES_Msg_Key, M)
ECC (EC_AuthServer_Pu, AES_Msg_Key)
signed with EC_Server_Pr
ECC (EC_AuthServer_Pu, AES_Msg_Key) signed with EC_Server_Pr)
EC_Client_Pr
Verify EC_Client_Pu
Decrypt AES_Msg_Key
ECC (EC_Client_Pu, AES_Msg_Key) signed with EC_AuthServer_Pr)
Decrypt AES_Msg_Key and then M

Other Security Practices

Also, Bamboo Apps uses the following common security practices:

- File extension checking;
- Session tokens, credentials are only delivered over HTTPS;
- Check password quality;
- Check good and wrong attempts;
- Multi-factor authentication;
- Out-of-channel notification of account lockouts and successful password changes;
- Brute force protection;
- Secure cookie session;

- Client-side and server-side validation rules;
- Sanitize filenames;
- Both server and client-generated keychain-pair for authentication device.

Configuration Management

As a key part of Software Configuration Management, we developed a critical checklist of tasks to be performed during black box security testing of a web application. Our actions checklist:

- Check for commonly used application and administrative URLs;
- Check for old, backup, and unreferenced files;
- Check HTTP methods supported and Cross Site Tracing (XST);
- Test file extensions handling;
- Test RIA cross-domain policy;
- Test for security HTTP headers (e.g. CSP, X-Frame-Options, HSTS);
- Test for policies (e.g. Flash, Silverlight, robots);
- Check for sensitive data in client-side code (e.g. API keys, credentials).

The last but not least part of automotive cybersecurity best practices is just around the corner, so keep an eye out if you are interested in finding out more.