Chapitre 1

La sécurité dans le monde de l'automobile

Ce travail fût présenté en partie à la conférence des sourds-muets unijambistes à Quibéron en avril 1994.

Maxime Ayrault

1.1 La révolution logicielle

Le logiciel embarqué est une des innovations clé dans le monde automobile. D'après l'article de Robert Charette (Charette 2009) paru dans IEEE Spectrum, la première voiture embarquant un logiciel était la Oldsmobile Toronado de General Motors en 1977. La Toronado avait une unité de contrôle électronique (ECU) qui gérait la synchronisation de l'allumage des bougies (spark timing). En 1978, General Motors proposait, en option sur ses Cadillacs, un ordinateur de bord qui pouvait afficher la vitesse, le niveau du réservoir d'essence, les informations sur l'état du véhicule. Ce logiciel s'exécutait sur une version modifiée du processeur Motorola 6802 et faisait 50.000 lignes de code. Depuis, de plus en plus de fonctions ont été réalisées par du logiciel. Afin de limiter le nombre de câbles dans une voiture, les capteurs ont été décentralisés et connectés à un réseau interne (réseau CAN). Le logiciel a également permis de créer de nouvelles fonctions pour les voitures.
Une voiture actuelle est maintenant devenue une plate-forme logiciel sur roues. Une voiture grand public contient entre 30 et 50 unités de contrôle électronique effectuant la gestion de multiples systèmes (voir 1.1).

Air bag	ABS	Système d'alarme
La climatisation	Le régulateur de vitesse	Le régime moteur
Les clignotants	Les feux	Le klaxon
La gestion des sièges	Le système de navigation	Le système audio
La pression des pneus	La gestion des portes et vitres	...

TABLE 1.1 – Système logiciel dans une voiture

En 2009, Alfred Katzenbach, le directeur des Technologies Informatique Chez Daimler, a annonçé que le système navigation et audio sur une Mercedes-Benz S-class contient plus 20 millions de lignes de code et que la voiture contienait pratiquement autant d'unité électronique qu'un Airbus A380 (si on exclut le système de divertissement en vol) (CHARETTE 2009). Les logiciels dans une voiture grossissent à une vitesse exponentielle en taille et en complexité. En 2010, certaines voitures avaient 10 millions de lignes de code. En dix ans, le volume du logiciel a augmenté par un facteur 15, pour environ 150 millions de lignes. Aujourd'hui, le Model S de Tesla est équipée d'un écran tactile de 17 pouces basé sur un système d'exploitation Linux qui contrôle quasiment toute les fonctions utilisateurs de la voiture. En fait, il n'y a plus que 2 boutons manuel non gérés par du logiciel sur un Model S ; le bouton pour les feux de détresse et le bouton de la boîte à gants.

Un effet secondaire est sur la complexité des fonctions proposées aux conducteurs. Il n'est pas rare que les manuels utilisateurs des voitures fassent maintenant plus de 500 pages pour expliquer l'ensemble des fonctions logiciel. D'un autre côté, on estime qu'un conducteur moyen n'utilise pas plus de 20% des fonctions implémentées par du logiciel. Les constructeurs automobiles commencent à se poser la question sur une limitation des fonctions réalisées par du logiciel. Par exemple, est-il vraiment nécessaire que la lumière du plafonnier d'une voiture s'éteigne de façon progressive ?

L'augmentation du logiciel a également d'importantes conséquences sur la façon d'entretenir et de réparer une voiture. On estime que plus de 50% des unités électroniques remplacées n'ont pas d'erreur (logiciel ou matériel). Le garagiste remplace fréquemment une pièce car il ne sait pas quelle est la cause principales de la panne. La principale activité des garagiste consiste à téécharger les nouvelles versions du logiciel. Sur les voitures Tesla, les mises à jour logiciel, qui incluent les corrections du logiciel et les nouvelles fonctionnalités sont téléchargées dans la voiture via le réseau cellulaire sans intervention d'un garagiste.

Aujourd'hui, le coût du logiciel et de l'électronique représente entre 35 à 40% du coût d'une voiture.

Todo list 1 *Introduire Automotive Open System Architecture (AUTOSAR). Une solution pour réduire le coût du logiciel en augmentant l'interopérabilité entre les constructeurs.*

L'introduction du logiciel a permis d'avoir des voitures plus sûrs et moins polluantes mais a également introduit une nouvelle menace ; la *sécurité informatique des voitures.*

1.2 Architecture voiture

Todo list 2 *Commencer a présenter les différentes couches logiciel.*

the following four stacks could become the basis for upcoming generations of cars in five to ten years :

- Time-driven stack. *In this domain, the controller is directly connected to a sensor or actuator while the systems have to support hard real-time requirements and low latency times ; resource scheduling is time based. This stack includes systems that reach the highest Automotive Safety Integrity Level classes, such as the classical Automotive Open System Architecture (AUTOSAR) domain.*
- Event- and time-driven stack. *This hybrid stack combines high-performance safety applications, for example, by supporting ADAS and HAD capability. Applications and peripherals are separated by the operating system, while applications are scheduled on a time base. Inside an application, scheduling of resources can be based on time or priority. The operating environment ensures that safety-critical applications run on isolated containers with clear separation from other applications within the car. A current example is adaptive AUTOSAR.*
- Event-driven stack.*This stack centers on the infotainment system, which is not safety critical. The applications are clearly separated from the peripherals, and resources are scheduled using best-effort or event-based scheduling. The stack contains visible and highly used functions that allow the user to interact with the vehicle, such as Android, Automotive Grade Linux, GENIVI, and QNX.*
- Cloud-based (off-board) stack. *The final stack covers and coordinates access to car data and functions from outside the car. The stack is responsible for communication, as well as safety and security checks of applications (authentication), and it establishes a defined car interface, including remote diagnostics.*

https://www.mckinsey.com/industries/automotive-and-assembly/our-insights/rethinking-car-software-and-electronics-architecture

https://www2.deloitte.com/us/en/insights/focus/future-of-mobility/pure-play-so
html

1.3 Principes de la sécurité informatique

La sécurité informatique est construite sur 3 piliers (CIA triad).

- La *confidentialité*. L'objectif consiste á éviter la divulgation d'informations sensibles et/ou de protéger les accès non authorisés ressources sensibles. Autrement dit, la confidentialité implique la protection des données (resp. ressources), en donnant accès à ceux qui sont autorisés à les lire (resp. à les utiliser) tout en interdisant aux autres d'apprendre quoi que ce soit sur son contenu (ou son existence).

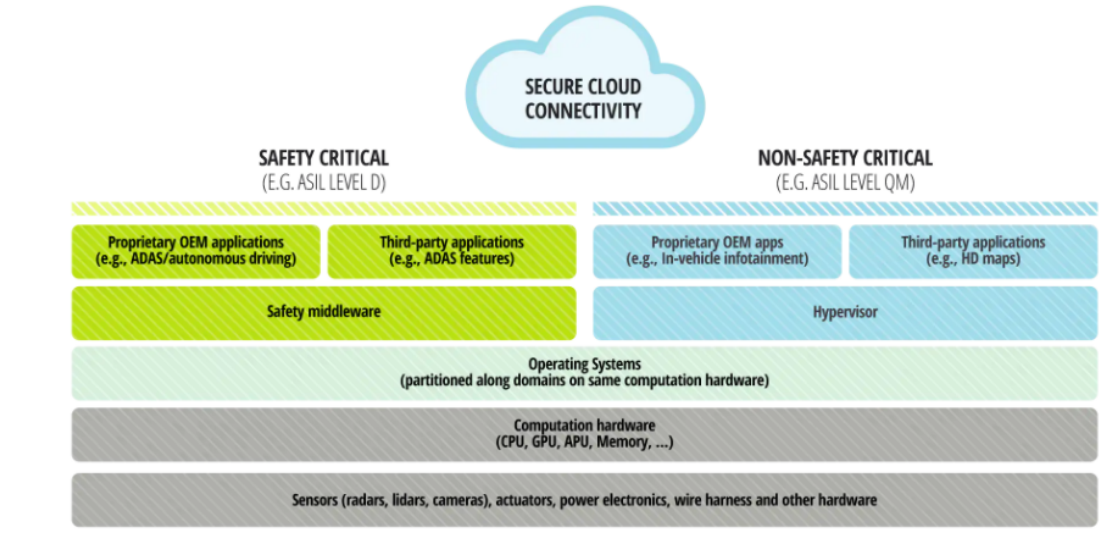

FIGURE 1.1 – Architecture générique

— L'*intégrité*. L'objectif est de protéger ou de minimiser les altérations non authorisées des données et/ou ressources sensibles. Autrement dit, l'intégrité implique de préserver le contenu des données et de détecter si une données ou une ressource a été modifiée depuis sa derniére écriture par une personne authorisée.

— La *disponibilité*. L'objectif est de maintenir le système opérationel pour rendre le service aux l'utilisateurs.

FIGURE 1.2 – Les pilers de la sécurité informatique

Les principales techniques utilisées pour garantir ces 3 propriétés sont :

— *Chiffrement (Encryption)* : La transformation de l'informations originale (appelé clair) à l'aide d'une clé de chiffrement, de telle sorte que les informations transformées (appelé chiffré) ne puissent être interprétées que par un autre utilisateur ayant connaissance de la clé de déchiffrement (qui peut, dans certains cas, être identique à la clé de chiffrement). Pour être en sécurité, un algorithme de chiffrement doit rendre extrêmement difficile pour quelqu'un de déterminer tout ou partie des informations clairs sans connaissance de la

clé de déchiffrement ou faiblesse de l'algorithme de chiffrement.

— *Authentification (Authentication)* : La détermination de l'identité qu'un sujet (personne, logiciel ou équipement). Cette détermination peut être effectuée par plusieurs moyens. Il est généralement basé sur une combinaison de quelque chose que la personne connaît (Something you know) comme un mot de passe ou un code PIN, quelque chose le sujet possède (Something you have) comme une carte à puce contenant des clés secrètes un téléphone pour recevoir un code, un passeport, ou quelque chose la personne est (Something you are) comme une empreinte digitale.

— *Contrôles d'accès (Access Control)* : Ensemble de règles et politiques de sécurité qui limitent l'accès aux informations aux sujets (personnes et/ou systèmes) ayant un *besoin d'en connaitre*. Ce besoin d'en connaitre est déterminé suite à l'authentification correcte du sujet, de par son identité ou rôle. Un ensemble de règles est pré-définie par le gestionnaire de la sécurité informatique en se basant sur la politique de sécurité.

— *Signature (Signature)* :

— *Responsabilité (Accountability)* :

— *Sensibilisation à la sécurité (Security awarness)* :

— *Sécurité physique (physical security)* : Mise en place de barrières physiques pour limiter l'accès aux ressources sensibles. Ces barriÃ¨res comprennent sont mutiples comme le gardiennage, la vidéo-surveillance, les serrures sur les armoires et les portes, les chambres fortes, l'utilisation de matÃ©riaux insonorisants, ou même la construction d'équipement renforcé (tempest) afin que les signaux électromagnétiques ne puissent pas entrer ou sortir.

— *Tolérance aux fautes (fault tolerance)* : Ensemble de techniques peuvent être utilisées pour garantir le systéme en opération.

	Confidentialité	Integrité	Disponibilité
Chiffrement	✓		
Authentification	✓	✓	
Contrôles d'accès	✓	✓	
Signature		✓	
Responsabilité	✓	✓	
Sensibilisation à la sécurité physique	✓	✓	✓
Sécurité physique	✓	✓	✓
Tolérance aux fautes			✓

TABLE 1.2 – Principales méthodes de la sécurité informatique

Dans le reste de la thèse, nous concentrerons sur la disponibilité des systèmes embarqués.

1.4 Le risque cybersécurité pour l'automobile

L'ajout de logiciels et de connectivité....

Todo list 3 *https://www.nytimes.com/2015/09/27/business/complex-car-software-html* *Fear of Hacking Andy Greenberg steered a 2014 white Jeep Cherokee down a highway in St. Louis, cruising along at 70 miles per hour. Miles away, two local hackers, Charlie Miller and Chris Valasek, sat on a leather couch at Mr. Miller's house, laptops open, ready to wreak havoc.*

As Mr. Greenberg sped along, both hands on the wheel, his ride began to go awry. First, the air-conditioning began blasting. Then an image of the hackers in tracksuits appeared on the digital display screen. Rap music began blaring at full volume, and Mr. Greenberg could not adjust the sound. The windshield wipers started and cleaning fluid sprayed, obstructing his view. Finally, the engine quit. Mr. Greenberg was on a highway with no shoulder. A big rig blew past, blaring its horn.

"I'm going to pull over," Mr. Greenberg said. " Because I have PTSD."

The episode was in fact a stunt orchestrated by the hackers and Mr. Greenberg, a writer for Wired magazine, to demonstrate the Jeep's very real vulnerabilities. The article appeared on July 21.

Days later, Fiat Chrysler, the maker of Jeep, announced a recall of 1.4 million vehicles to fix the flaws the hackers had identified - the first known recall intended to address a possible hacking threat.