ARM TrustZone을 이용한 AUTOSAR CSM 설계

손희승○ 조진성
경희대학교 컴퓨터공학과
toddlf95@khu.ac.kr, chojs@khu.ac.kr

Design of AUTOSAR CSM by using ARM TrustZone

Heeseung Son○ Jinsung Cho
Department of Computer Science and Engineering, KyungHee University

요 약

최근 자동차 시스템의 전자화가 이뤄짐에 따라, 자동차 ECU (Electronic Control Unit)의 펌웨어는 AUTOSAR (AUTomobile Open System ARchitecture) 표준을 따라 개발된다. 그러나 현재 대부분 자동차의 ECU 간 통신에는 암호화 표준이 적용되지 않아 보안이 매우 취약하다. AUTOSAR는 이러한 보안 이슈를 해결하기 위해 AUTOSAR CSM (Crypto Service Manager) 표준을 제공하였으나, Firmware Modification Attack (FMA)등의 Hardware 취약점 공격에는 여전히 노출되어 있다. 이에 본 논문에서는 ARM에서 제공하는 ARM TrustZone HSM을 이용하여 Hardware Security 공격에도 안전한 AUTOSAR CSM API를 설계한다.

1. 서 론

최근 통신기술 및 전자장비의 발전으로 인해 수많은 IoT 디바이스가 개발되어 출시됐다. 이런 흐름에 맞춰 자동차 또한 다양한 전자장비를 탑재하고 있으며, 이에 따라 변속 제어, 차체 제어, 크루즈 모드, ABS (Auto Breaking System) 등 여러 자동차 제어 기능이 추가됐다. 또한, 최근 출시된 차량은 자동차 ECU에 통신 모듈을 장착하여, 모바일 어플리케이션으로 차의 상태를 확인하고 제어할 수 있다. 이런 기능은 기존 자동차의 수동적이고 기계적인 제어에서 능동적이고 자동적이며 전자적인 제어로의 변화를 의미한다.

그러나 자동차의 전자화 비율이 높아지고 전자 제어에 의존할수록, 전자장비 해킹 및 오작동으로 인한 위험 또한 증가한다. 실제로 미라이 봇넷을 통해 보안을 고려하지 않은 IoT 디바이스의 위험성이 증명됐다. 이에 국/내외 자동차 회사는 ECU 소프트웨어 구조 표준인 AUTOSAR에 crypto service 표준을 지정하고, 이를 AUTOSAR CSM이라 정의했다.

그러나 AUTOSAR CSM 표준은 software level 관점에서의 신뢰성 보장을 다루고 있으며, hardware level을 고려하고 있지 않는 문제를 가진다. 즉, AUTOSAR CSM을 통해 통신 암호화 및 데이터 암호화와 같은 안전한 기능을 사용할 수 있지만, 해커가 flash memory에 access하여 암호 키를 탈취하거나 firmware를 변조하는 공격에는 신뢰성을 보장하지 못한다.

본 논문에서는 이러한 문제를 해결하기 위해 HSM을 사용한 안전한 AUTOSAR CSM의 설계를 목적으로 한다. 이를 위해 ARM의 TrustZone을 이용하여 민감 데이터의 실행환경을 격리하는 방안을 제안한다.

본 논문은 2절에서 ARM TrustZone과 AUTOSAR CSM에 대해 조사한다. 2절에서 조사한 내용을 바탕으로 3절에서는 ARM TrustZone을 이용한 AUTOSAR CSM API를 설계하고, 4절에서는 결론 및 향후 연구 방향을 제시한다.

이 논문은 교육부 및 한국연구재단의 기초연구사업(NRF-2017R1D1A1B04035914)과 과학기술정보통신부 및 정보통신기획평가원의 SW중심대학 사업(2017-0-00093)의 지원으로 수행된 연구결과임.

2. 연구배경 및 관련연구

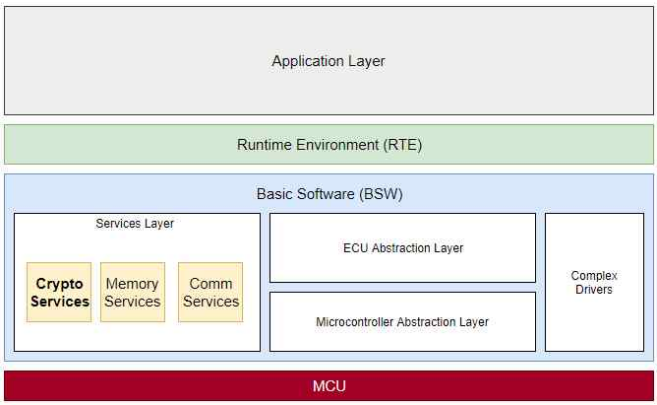
2.1. ARM TrustZone

ARM TrustZone은 시스템 리소스의 실행환경을 Secure 영역과 Non-Secure 영역으로 격리하여 Secure 영역의 안전을 보장한다 [1]. 이를 위해 Non-Secure에서 동작하는 kernel과 user application은 Secure 영역에 접근하지 못하도록 하드웨어 레벨 권한이 설정되어 있다. 만약 Non-Secure 영역에서 Secure 영역의 기능을 요청할 때는 NSC (Non-Secure Callable)라 불리는 함수를 통해 접근해야 한다. 반면에, Secure 영역에서 동작하는 Secure Application은 Secure 영역 뿐만 아니라 Non-Secure 영역의 메모리 또한 접근할 수 있다 [2]. 따라서, 보안에 민감한 소프트웨어를 Secure 영역에서 처리함으로써 Non-Secure 영역의 소프트웨어가 해킹 등으로 인한 오작동이 발생하더라도 보호할 수 있다.

2.2. AUTOSAR CSM

AUTOSAR CSM은 AUTOSAR의 application 모듈에서 crypto기능을 쉽게 사용할 수 있도록 설계된 표준 API이

다. AUTOSAR 표준에서 제공하는 Layered Software Architecture 문서에서는 [그림 1]과 같이 AUTOSAR의 전체 구조를 제시하고 있다 [3]. AUTOSAR에서 CSM의 위치는 운영체제의 역할을 하는 BSW(Basic SoftWare)의 Services Layer에 존재하므로, 일반 Application Layer에서 AUTOSAR CSM을 이용하기 위해서는 RTE (RunTime Environment)라 불리는 미들웨어를 통해 간접적으로 호



[그림 1] AUTOSAR 전체구조

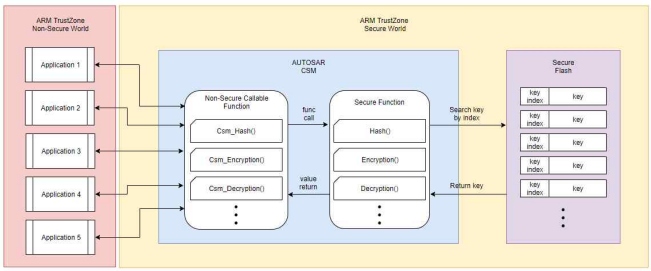
출해야만 한다. AUTOSAR CSM 표준에는 hash, random number generator, key generator, symmetric enc/dec, asymmetric enc/dec등의 기본적인 암호화 관련 API 뿐만 아니라, data signature, certificate, MAC에 대한 generate 및 verify 등의 향상된 기능도 제공한다 [4]. 또한, 암호화 API application에서 외부 네트워크 혹은 다른 ECU와 CAN통신을 할 때 혹은 보안에 민감한 data를 저장할 때 암호화하여 처리할 것을 권장하고 있다. [표 1]은 AUTOSAR CSM의 대표적인 API에 대한 설명이다.

3. ARM TrustZone을 이용한 AUTOSAR CSM 설계

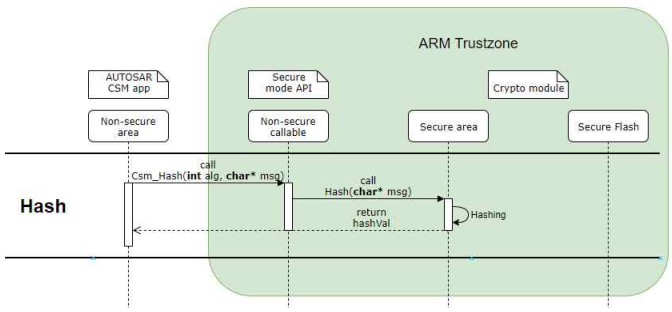
본 절에서는 ARM TrustZone을 이용하여 AUTOSAR CSM 설계를 제안한다. 제안하는 ARM TrustZone 기반의 AUTOSAR CSM은 ARM TrustZone의 Non-Secure영역과 Secure영역으로 분리하여 user application이 보안에 민감한 CSM 및 secure data에 직접적으로 접근하지 못하도록 한다. TrustZone CSM에서는 [그림 2]와 같이 Non-Secure 영역에 application들을 위치시키고, Secure영역에 CSM 및 Secure Flash를 위치시킨다. 이 때, NSC에는 [표 1]과 같이 application과 상호작용할 CSM의 API들을 배치하고, Secure영역에는 crypto 알고리즘 및 암호 키를 저장할 Secure Flash를 배치한다. Secure Flash는 암호 키와 같은 보안에 민감한 데이터들을 key-value방식으로 저장하는 저장장치로써, 기본적으로 ARM MCU 내부 flash에 설정하고, 해당 메모리 주소를 Secure영역으로 할당해 보호되도록 한다. 만일 Secure Flash가 추가적인 용량을 필요로 할 경우, Secure Flash를 암호화할 키만 MCU 내부 Secure 영역의 flash에 저장하고, 외부 flash에는 해당 암호키를 이용해 암호화하여 저장한다. 이를 통해 Non-Secure영역의 application들은 NSC를 통해야만 crypto 알고리즘을 사용할 수 있으며, Key를 저장하는 Secure Flash에는 접근이 불가능하다. TrustZone CSM의 기본적인 흐름과 AUTOSAR CSM의 각 응용 API들의 기저가 되는

[표 1] AUTOSAR CSM API example

Service name	Description
Csm_Init()	CSM모듈 초기화
Csm_Hash()	해시 연산 수행
Csm_MacGenerate()	MAC 생성
Csm_MacVerify()	MAC 검증
Csm_Encrypt()	암호화
Csm_Decrypt()	복호화
Csm_SignatureGenerate()	전자서명 생성
Csm_SignatureVerify()	전자서명 검증
Csm_RandomGenerate()	랜덤숫자 생성
Csm_KeyElementGet()	키 인덱스로 키 반환
Csm_KeyGenerate()	키 생성



[그림 2] ARM TrustZone를 이용한 AUTOSAR CSM overview

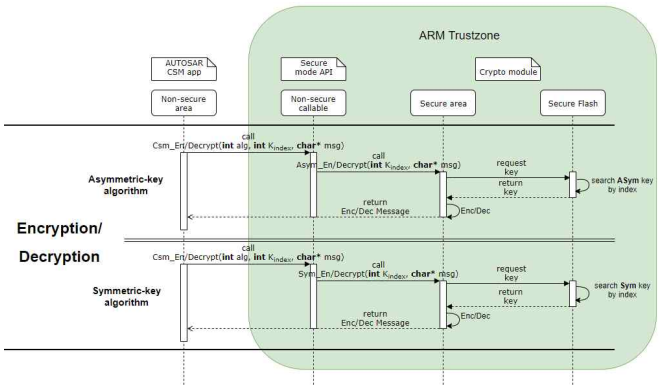


[그림 3] ARM TrustZone를 이용한 AUTOSAR CSM Hash API 시퀀스 다이어그램

API는 다음과 같다.

• **Basic Flow**: Application은 Secure영역의 암호화 알고리즘을 사용하기 위해 NSC에 등록된 CSM API를 호출하며, 암호 키를 사용하는 API의 경우 매개변수에 사용할 key의 index를 전달한다. application으로부터 호출된 CSM API는 Secure 영역의 암호화 알고리즘들을 호출하고, 호출된 알고리즘 함수는 전달된 data에 대해 연산을 수행

하고 NSC로 반환한다. 이 때, key가 필요한 알고리즘의 경우 매개변수로 전달된 key index를 사용하여 Secure Flash로부터 필요한 key를 찾을 수 있다. 마지막으로, 연산을 마친 NSC는 Non-Secure영역에 쓰기 작업을 할 수 있는 권한을 가지고 있으므로 application memory 영역에 반환값을 write함으로써 결과값을 전달한다.



[그림 4] ARM TrustZone를 이용한 AUTOSAR CSM enc/decrypt API 시퀀스 다이어그램

• **Hash**: AUTOSAR CSM에서는 Csm_Hash() API를 이용하여 Application에 Hash값을 전달한다. 본 논문에서는 [그림 3]과 같이 Non-Secure 영역의 application과 상호작용하는 Csm_Hash() 함수를 ARM TrustZone의 NSC(Non-Secure callable)에 위치한다. callable 함수는 application으로부터 전달된 매개변수 alg값을 이용하여 Secure영역에 위치한 alg 알고리즘의 hash함수를 호출하고, 호출된 Secure영역의 hash함수는 전달된 msg를 hashing한 뒤 Non-Secure 영역의 메모리에 write하여 application에 결과값을 전달하도록 설계하였다.

• **Encryption/Decryption**: AUTOSAR CSM에서는 데이터의 암호화/복호화를 위해서 Csm_Encrypt(), Csm_Decrypt() 두 가지의 API를 사용하여 데이터의 암/복호화를 진행한다. 본 논문에서는 위의 두 API를 NSC 영역에 위치시켜 사용한다. 해당 API는 [그림 4]와 같이 Non-Secure 영역의 application으로부터 매개변수 alg에 사용할 알고리즘(비대칭 키, 대칭 키 암호 알고리즘 등)을 전달받고, 매개변수 K_{index}로 암/복호화를 진행할 key의 index를 전달받는다. 매개변수를 전달받은 API는 alg 매개변수에 따라 Secure 영역의 암호화 알고리즘 함수를 호출하고, 호출된 Secure 영역의 함수는 K_{index}를 통해 Secure Flash에 저장된 key값을 읽어들여 encrypt/decrypt연산을 수행한 뒤 Non-Secure 영역의 지정된 메모리에 write하여 application에 결과값을 전달하도록 설계하였다.

• **Key generate**: AUTOSAR CSM에서는 암호화/복호화 알고리즘에 사용될 Key를 만들기 위해서 Csm_KeyGenerate() API를 사용한다. 본 논문에서는 해당 API를 NSC 영역에 위치시켜 사용한다. 해당 API는 Non-Secure 영역의 application에서 매개변수 alg에 해당 Key를 사용할 암호 알고리즘을 전달받는다. 매개변수를 전달받은 API는 alg 매개변수에 따라 Secure영역의 Keygen()함수를 호출하고, 호출된 Keygen()함수는 키를 생성한 뒤 사용되지 않은 K_{index}와 생성한 키를 한 쌍으로 묶어 Secure Flash에 저장한다. K_{index}와 생성한 키의 저장이 확인되면 Keygen() 함수는 Non-Secure영역의 지정된 메모리에 K_{index}를 write하여 application에 결과값을 전달하도록 설계하였다.

• **Key load**: AUTOSAR CSM에서는 비대칭키의 키교환 등을 위해 key를 Secure Flash로부터 읽어올 때 Csm_KeyElementGet() API를 사용한다. 본 논문에서는 해당 API를 NSC영역에 위치하여 사용한다. 해당 API는 Non-Secure 영역의 application으로부터 매개변수 K_{index}에 읽어올 key의 index를 전달받는다. 매개변수를 전달받은 API함수는 Secure영역의 getKey()함수를 호출하고, 호출된 getKey() 함수는 Secure Flash로부터 K_{index}와 쌍을 이루는 key를 읽어와 Non-Secure영역의 지정된 메모리에 write하여 key값을 전달한다.

• **Random number generate**: AUTOSAR CSM에서는 암호화 통신시 nonce등의 사용을 위해 random number가 필요할 때 Csm_RandomGenerate() API를 사용한다. 본 논문에서는 해당 API를 NSC영역에 위치하여 사용한다. 해당 API는 Non-Secure영역의 application에서 호출되며, 호출된 API함수는 Secure영역의 RandomGenerate() 함수를 호출하고, 호출된 RandomGenerate() 함수는 random value를 생성하여 Non-Secure영역의 지정된 메모리에 write하여 application에 전달한다.

4. 결론 및 향후연구

본 논문에서는 AUTOSAR CSM에 ARM TrsutZone을 이용하여 user application 영역과 secure application 영역을 물리적으로 분리함으로써 software 뿐만 아니라 hardware 적으로도 안전한 AUTOSAR CSM 설계를 제안하였다. 이를 통해, 신뢰성 높은 자동차 펌웨어를 개발할 수 있을 것으로 기대된다.

향후 연구로는 CSM API뿐만 아니라 ARM TrustZone을 이용한 secure boot, remote attestation, local attestation등의 기능을 개발할 계획이다.

참고문헌

[1] ARM Limited, “ARM Platform Security Architecture Trusted Base System Architecture for ARMv6-M, ARMv7-M and ARMv8-M 1.0 Beta-1”, 2019.
[2] Junyoung Jung, Jinsung Cho, “Design of a Secure Platform for IoT Device based on ARM PSA”, KSC, 2019
[3] AUTOSAR, “Layered Software Architecture”, 2017
[4] AUTOSAR, “Specification of Crypto Service Manager”, 2017