(19) 대한민국특허청(KR)
(12) 공개특허공보(A)

(11) 공개번호 10-2020-0076217
(43) 공개일자 2020년06월29일

(51) 국제특허분류(Int. Cl.)
H04L 29/06 (2006.01) *H04L 12/24* (2006.01)
H04L 12/40 (2006.01)
(52) CPC특허분류
H04L 63/1458 (2013.01)
H04L 12/40104 (2013.01)
(21) 출원번호 10-2018-0165100
(22) 출원일자 2018년12월19일
심사청구일자 2018년12월19일

(71) 출원인
한림대학교 산학협력단
강원도 춘천시 한림대학길 1, 한림대학교(옥천동)
(72) 발명자
조효진
강원도 춘천시 한림대학길 1, 한림대학교 공학관 1305호
김형훈
서울특별시 강북구 덕릉로 191-10, 2층
(74) 대리인
김남혁

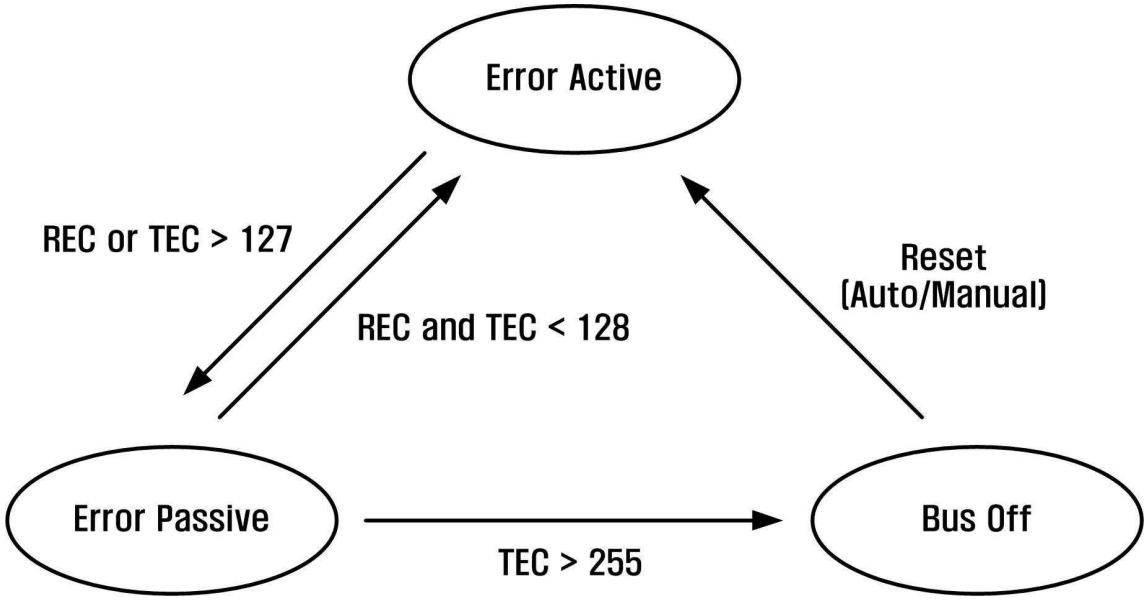
전체 청구항 수 : 총 10 항

(54) 발명의 명칭 **공격 메시지 재전송 시간을 예측하는 CAN 네트워크에 대한 메시지플러딩 공격 완화방법**

(57) 요 약

컴퓨터에 의하여 수행되는 방법에 있어서, 일반 CAN 메시지들의 전송 스케쥴에 기반하여, 일반 CAN 메시지와 메시지 플러딩 공격에 이용된 CAN 메시지를 식별하는, 공격메시지 식별단계, 공격 노드가 메시지 플러딩 공격을 위해 다수의 CAN 메시지를 포함하는 공격 메시지를 전송할 때, 도미넌트(Dominant) 비트를 생성하여 상기 공격 노드에 CAN 에러 프레임 생성을 유도하는, 도미넌트 비트 생성단계 및 상기 도미넌트 비트로 인해 상기 공격 노드가 공격 메시지 전송을 실패하는 경우, 상기 공격 메시지를 재전송하기까지의 시간을 예측하는, 공격 메시지 재전송 시간($T_{retrans}$) 예측단계를 포함하는, CAN 네트워크에 대한 메시지플러딩 공격을 완화하는 방법이 개시된다.

대 표 도 - 도1



(52) CPC특허분류
H04L 12/4015 (2013.01)
H04L 41/147 (2013.01)
H04L 63/1408 (2013.01)
H04L 2012/40273 (2013.01)

이 발명을 지원한 국가연구개발사업
과제고유번호 2018-0-00216
부처명 정보통신기술진흥센터
연구관리전문기관 한림대학교 산학협력단
연구사업명 SW중심대학
연구과제명 SW중심대학(한림대학교)
기 여 율 1/2
주관기관 한림대학교 산학협력단
연구기간 2018.09.01 ~ 2018.12.31
이 발명을 지원한 국가연구개발사업
과제고유번호 H20180758
부처명 한국연구재단
연구관리전문기관 한림대학교 산학협력단
연구사업명 생애 첫 연구사업
연구과제명 차량 해킹 공격에 강건한 스마트 자동차 시스템을 위한 인증, 도스 미티게이터, 그리고 로깅 기술의 연구
기 여 율 1/2
주관기관 한림대학교 산학협력단
연구기간 2018.09.01 ~ 2019.02.28

명 세 서

청구범위

청구항 1

컴퓨터에 의하여 수행되는 방법에 있어서,

일반 CAN 메시지들의 전송 스케쥴에 기반하여, 일반 CAN 메시지와 메시지 플러딩 공격에 이용된 CAN 메시지를 식별하는, 공격메시지 식별단계;

공격 노드가 메시지 플러딩 공격을 위해 다수의 CAN 메시지를 포함하는 공격 메시지를 전송할 때, 도미넌트(Dominant) 비트를 생성하여 상기 공격 노드에 CAN 에러 프레임 생성을 유도하는, 도미넌트 비트 생성단계; 및

상기 도미넌트 비트로 인해 상기 공격 노드가 공격 메시지 전송을 실패하는 경우, 상기 공격 메시지를 재전송하기까지의 시간을 예측하는, 공격 메시지 재전송 시간($T_{retrans}$) 예측단계; 를 포함하는,

CAN 네트워크에 대한 메시지플러딩 공격을 완화하는 방법.

청구항 2

제1 항에 있어서,

상기 공격 메시지 재전송 시간 예측단계는,

CAN의 에러 관리 절차를 획득하는 단계; 및

상기 획득된 에러 관리 절차에 기초하여 상기 공격 메시지의 재전송 시간을 예측하되, 상기 도미넌트 비트로 인해 상기 공격 노드가 에러 프레임을 전송한 시점 및 상기 공격노드의 TEC(Transmission Error Counter)에 기초하여 상기 공격 메시지의 재전송 시간을 예측하는, 단계; 를 포함하는,

CAN 네트워크에 대한 메시지플러딩 공격을 완화하는 방법.

청구항 3

제1 항에 있어서,

공격 메시지 및 에러 프레임으로 인하여 일반 CAN 메시지의 전송지연 발생시 상기 일반 CAN 메시지의 전송지연을 예측하는, 일반 CAN 메시지 전송지연 예측단계; 를 더 포함하는,

CAN 네트워크에 대한 메시지플러딩 공격을 완화하는 방법.

청구항 4

제3 항에 있어서,

상기 일반 CAN 메시지 전송지연 예측단계는,

상기 일반 CAN 메시지의 전송지연의 최악의 시간(WT_{delay})을 예측하는 단계; 를 포함하는,

CAN 네트워크에 대한 메시지플러딩 공격을 완화하는 방법.

청구항 5

제4 항에 있어서,

상기 최악의 시간을 예측하는 단계는,

아래 수학식에 기초하여 상기 최악의 시간을 예측하는 단계; 를 포함하고,

$$WT_{delay} = T_{attack} + T_{retrans} + T_{error}$$

상기 T_{attack}은 상기 공격 메시지로 인한 지연시간을 의미하고, 상기 T_{error}는 상기 에러 프레임으로 인한 지연시간을 의미하는,

CAN 네트워크에 대한 메시지플러딩 공격을 완화하는 방법.

청구항 6

제3 항에 있어서,

상기 공격 노드로부터 발생하는 공격 메시지의 패턴을 획득하는 단계;

데이터베이스로부터 상기 패턴에 대응하는 공격수단에 대한 정보를 획득하는 단계;

상기 획득된 정보로부터 하나 이상의 파라미터를 추출하는 단계; 및

상기 추출된 파라미터에 기반하여 상기 공격 메시지 재전송 시간 및 상기 일반 CAN 메시지 전송지연 예측결과를 조정하는 단계; 를 더 포함하는,

CAN 네트워크에 대한 메시지플러딩 공격을 완화하는 방법.

청구항 7

제1 항에 있어서,

상기 공격 노드가 Bus off 되는 경우, 상기 공격 노드에 대응하는 예비 노드를 상기 CAN 네트워크에 연결하는 단계; 및

상기 예비 노드가 상기 공격 노드의 역할을 수행하도록 제어하는 단계; 를 더 포함하는,

CAN 네트워크에 대한 메시지플러딩 공격을 완화하는 방법.

청구항 8

컴퓨터에 의하여 수행되는 방법에 있어서,

일반 CAN 메시지들의 전송 스케쥴에 기반하여, 일반 CAN 메시지와 메시지 플러딩 공격에 이용된 CAN 메시지를 식별하는, 공격메시지 식별단계;

공격 노드가 메시지 플러딩 공격을 위해 다수의 CAN 메시지를 포함하는 공격 메시지를 전송할 때, 도미넌트(Dominant) 비트를 생성하여 상기 공격 노드에 CAN 에러 프레임 생성을 유도하는, 도미넌트 비트 생성단계; 및

공격 메시지 및 에러 프레임으로 인하여 일반 CAN 메시지의 전송지연 발생시 상기 일반 CAN 메시지의 전송지연을 예측하는, 일반 CAN 메시지 전송지연 예측단계; 를 포함하는,

CAN 네트워크에 대한 메시지플러딩 공격을 완화하는 방법.

청구항 9

하나 이상의 인스트럭션을 저장하는 메모리; 및

상기 메모리에 저장된 상기 하나 이상의 인스트럭션을 실행하는 프로세서를 포함하고,

상기 프로세서는 상기 하나 이상의 인스트럭션을 실행함으로써,

제1 항의 방법을 수행하는, 장치.

청구항 10

하드웨어인 컴퓨터와 결합되어, 제1 항의 방법을 수행할 수 있도록 컴퓨터에서 독출가능한 기록매체에 저장된 컴퓨터프로그램.

발명의 설명

기 술 분 야

[0001] 본 발명은 공격 메시지 재전송 시간을 예측하는 CAN 네트워크에 대한 메시지플러딩 공격 완화방법에 관한 것이다.

배 경 기 술

[0002] 최근에는 자동차의 전자식 제어를 넘어 오토파일럿 등의 기초적인 자율주행 지원 모듈들이 활발하게 개발되고 있다. 예를 들어, 완전한 자율주행 기술(Full Self-Driving Automation)을 위해 LiDAR (거리 측정/사물인지 센서)등의 센서 기술들과 외부 통신기술(V2X: Vehicle to Everything)을 지원하는 자동차 IT 융합 기술들이 개발되고 있다.

[0003] 이를 위하여, 자동차 내부의 제어유닛(Electronic Control Unit, ECU)들은 자동차 내부 네트워크(Controller Area Network (CAN), 차량용 Ethernet)를 통해 정보를 공유하고, 자동차를 제어한다.

[0004] 이와 같이 자동차와 IT의 융합기술이 발전됨에 따라 자동차의 제어 네트워크에 접근할 수 있는 외부 접점들이 빠르게 증가하고 있다. 이러한 외부 접점들을 이용한 자동차 해킹가능성에 대한 연구들이 많이 수행되었고, 일예로 Jeep에서는 텔레매틱스 시스템의 해킹 결과로 인해 140만대의 차를 리콜한 사례가 있다.

[0005] 일 예로, 자동차 내부 네트워크를 보호하기 위해 인증, 침임탐지 시스템, 그리고 차량용 방화벽 기술 등이 연구되고 있다. 하지만, 원격 자동차 진단, 원격 ECU 업데이트 및 외부와의 통신을 이용하는 자율주행자동차 기술 등의 발달은 자동차 제어네트워크와 외부 네트워크 간의 접점들을 증가시키고, 이에 따라 공격자가 자동차 제어 네트워크의 다양한 외부 접점들을 이용하여 기존의 자동차 보안 기술들을 우회할 수 있는 가능성이 존재한다.

[0006] 특히, CAN 장애를 발생시킬 수 있는 메시지 플러딩 공격의 경우 내부 네트워크의 메시지 빈도수 변화나 네트워크 트래픽 증가를 통해 이를 쉽게 탐지할 수 있으나, 메시지플러딩 공격이 자동차 내부 네트워크의 장애(e.g., 전송지연) 를 일으키는 것은 막기가 어렵다. 따라서, 메시지플러딩 공격에 CAN에 발생했을 경우, 이를 완화하는 방법에 대한 연구가 필요하다.

선행기술문헌

특허문헌

[0007] (특허문헌 0001) 등록특허공보 제10-1825711호, 2018.01.30 등록

발명의 내용

해결하려는 과제

[0008] 본 발명이 해결하고자 하는 과제는 공격 메시지 재전송 시간을 예측하는 CAN 네트워크에 대한 메시지플러딩 공격 완화방법을 제공하는 것이다.

[0009] 본 발명이 해결하고자 하는 과제들은 이상에서 언급된 과제로 제한되지 않으며, 언급되지 않은 또 다른 과제들은 아래의 기재로부터 통상의 기술자에게 명확하게 이해될 수 있을 것이다.

과제의 해결 수단

[0010] 상술한 과제를 해결하기 위한 본 발명의 일 면에 따른 CAN 네트워크에 대한 메시지플러딩 공격을 완화하는 방법은, 일반 CAN 메시지들의 전송 스케쥴에 기반하여, 일반 CAN 메시지와 메시지 플러딩 공격에 이용된 CAN 메시지를 식별하는, 공격메시지 식별단계, 공격 노드가 메시지 플러딩 공격을 위해 다수의 CAN 메시지를 포함하는 공격 메시지를 전송할 때, 도미넌트(Dominant) 비트를 생성하여 상기 공격 노드에 CAN 에러 프레임 생성을 유도하는, 도미넌트 비트 생성단계 및 상기 도미넌트 비트로 인해 상기 공격 노드가 공격 메시지 전송을 실패하는 경우, 상기 공격 메시지를 재전송하기까지의 시간을 예측하는, 공격 메시지 재전송 시간($T_{retrans}$) 예측단계를 포함한다.

[0011] 또한, 상기 공격 메시지 재전송 시간 예측단계는, CAN의 에러 관리 절차를 획득하는 단계 및 상기 획득된 에러 관리 절차에 기초하여 상기 공격 메시지의 재전송 시간을 예측하되, 상기 도미넌트 비트로 인해 상기 공격 노드가 에러 프레임을 전송한 시점 및 상기 공격노드의 TEC(Transmission Error Counter)에 기초하여 상기 공격 메

시지의 재전송 시간을 예측하는, 단계를 포함할 수 있다.

[0012] 또한, 공격 메시지 및 에러 프레임으로 인하여 일반 CAN 메시지의 전송지연 발생시 상기 일반 CAN 메시지의 전송지연을 예측하는, 일반 CAN 메시지 전송지연 예측단계를 더 포함할 수 있다.

[0013] 또한, 상기 일반 CAN 메시지 전송지연 예측단계는, 상기 일반 CAN 메시지의 전송지연의 최악의 시간(WT_{delay})을 예측하는 단계를 포함할 수 있다.

[0014] 또한, 상기 최악의 시간을 예측하는 단계는, 아래 수학식에 기초하여 상기 최악의 시간을 예측하는 단계를 포함하고,

[0015] $$WT_{delay} = T_{attack} + T_{retrans} + T_{error}$$

[0016] 상기 T_{attack}은 상기 공격 메시지로 인한 지연시간을 의미하고, 상기 T_{error}는 상기 에러 프레임으로 인한 지연시간을 의미할 수 있다.

[0017] 또한, 상기 공격 노드로부터 발생하는 공격 메시지의 패턴을 획득하는 단계, 데이터베이스로부터 상기 패턴에 대응하는 공격수단에 대한 정보를 획득하는 단계, 상기 획득된 정보로부터 하나 이상의 파라미터를 추출하는 단계 및 상기 추출된 파라미터에 기반하여 상기 공격 메시지 재전송 시간 및 상기 일반 CAN 메시지 전송지연 예측결과를 조정하는 단계를 더 포함할 수 있다.

[0018] 또한, 상기 공격 노드가 Bus off 되는 경우, 상기 공격 노드에 대응하는 예비 노드를 상기 CAN 네트워크에 연결하는 단계 및 상기 예비 노드가 상기 공격 노드의 역할을 수행하도록 제어하는 단계를 더 포함할 수 있다.

[0019] 상술한 과제를 해결하기 위한 본 발명의 일 면에 따른 CAN 네트워크에 대한 메시지플러딩 공격을 완화하는 방법은, 일반 CAN 메시지들의 전송 스케쥴에 기반하여, 일반 CAN 메시지와 메시지 플러딩 공격에 이용된 CAN 메시지를 식별하는, 공격메시지 식별단계, 공격 노드가 메시지 플러딩 공격을 위해 다수의 CAN 메시지를 포함하는 공격 메시지를 전송할 때, 도미넌트(Dominant) 비트를 생성하여 상기 공격 노드에 CAN 에러 프레임 생성을 유도하는, 도미넌트 비트 생성단계 및 공격 메시지 및 에러 프레임으로 인하여 일반 CAN 메시지의 전송지연 발생시 상기 일반 CAN 메시지의 전송지연을 예측하는, 일반 CAN 메시지 전송지연 예측단계를 포함할 수 있다.

[0020] 상술한 과제를 해결하기 위한 본 발명의 일 면에 따른 CAN 네트워크에 대한 메시지플러딩 공격을 완화하는 장치는, 하나 이상의 인스트럭션을 저장하는 메모리 및 상기 메모리에 저장된 상기 하나 이상의 인스트럭션을 실행하는 프로세서를 포함하고, 상기 프로세서는 상기 하나 이상의 인스트럭션을 실행함으로써, 일반 CAN 메시지들의 전송 스케쥴에 기반하여, 일반 CAN 메시지와 메시지 플러딩 공격에 이용된 CAN 메시지를 식별하는, 공격메시지 식별단계, 공격 노드가 메시지 플러딩 공격을 위해 다수의 CAN 메시지를 포함하는 공격 메시지를 전송할 때, 도미넌트(Dominant) 비트를 생성하여 상기 공격 노드에 CAN 에러 프레임 생성을 유도하는, 도미넌트 비트 생성단계 및 상기 도미넌트 비트로 인해 상기 공격 노드가 공격 메시지 전송을 실패하는 경우, 상기 공격 메시지를 재전송하기까지의 시간을 예측하는, 공격 메시지 재전송 시간($T_{retrans}$) 예측단계를 포함하는 방법을 수행할 수 있다.

[0021] 상술한 과제를 해결하기 위한 본 발명의 일 면에 따른 CAN 네트워크에 대한 메시지플러딩 공격을 완화하는 프로그램은, 하드웨어인 컴퓨터와 결합되어, 개시된 실시 예에 따른 방법을 수행할 수 있도록 컴퓨터에서 독출가능한 기록매체에 저장된다.

[0022] 본 발명의 기타 구체적인 사항들은 상세한 설명 및 도면들에 포함되어 있다.

발명의 효과

[0023] 개시된 실시 예에 따르면, CAN 네트워크에 대한 메시지 플러딩 공격을 완화시킬 수 있으면서, 이에 따라 공격 노드가 아닌 다른 ECU들의 통신이 정상적으로 수행될 수 있도록 하는 효과가 있다.

[0024] 본 발명의 효과들은 이상에서 언급된 효과로 제한되지 않으며, 언급되지 않은 또 다른 효과들은 아래의 기재로부터 통상의 기술자에게 명확하게 이해될 수 있을 것이다.

도면의 간단한 설명

[0025] 도 1은 일 실시 예에 따른 ECU의 CAN 에러 관리 방법을 도시한 도면이다.

도 2는 일반적인 CAN-bus에 메시지 플러딩 공격이 발생한 일 예를 도시한 도면이다.

도 3은 개시된 실시 예에 따른 프레임워크가 적용된 CAN-bus에 메시지 플러딩 공격이 발생한 일 예를 도시한 도면이다.

도 4는 일 실시 예에 따라 CAN 네트워크에 대한 메시지플러딩 공격을 완화하는 방법을 도시한 흐름도이다.

도 5는 다른 일 실시 예에 따른 메시지플러딩 공격 완화방법을 도시한 흐름도이다.

도 6은 일 실시 예에 따른 장치의 구성도이다.

발명을 실시하기 위한 구체적인 내용

[0026] 본 발명의 이점 및 특징, 그리고 그것들을 달성하는 방법은 첨부되는 도면과 함께 상세하게 후술되어 있는 실시예들을 참조하면 명확해질 것이다. 그러나, 본 발명은 이하에서 개시되는 실시예들에 제한되는 것이 아니라 서로 다른 다양한 형태로 구현될 수 있으며, 단지 본 실시예들은 본 발명의 개시가 완전하도록 하고, 본 발명이 속하는 기술 분야의 통상의 기술자에게 본 발명의 범주를 완전하게 알려주기 위해 제공되는 것이며, 본 발명은 청구항의 범주에 의해 정의될 뿐이다.

[0027] 본 명세서에서 사용된 용어는 실시예들을 설명하기 위한 것이며 본 발명을 제한하고자 하는 것은 아니다. 본 명세서에서, 단수형은 문구에서 특별히 언급하지 않는 한 복수형도 포함한다. 명세서에서 사용되는 "포함한다(comprises)" 및/또는 "포함하는(comprising)"은 언급된 구성요소 외에 하나 이상의 다른 구성요소의 존재 또는 추가를 배제하지 않는다. 명세서 전체에 걸쳐 동일한 도면 부호는 동일한 구성 요소를 지칭하며, "및/또는"은 언급된 구성요소들의 각각 및 하나 이상의 모든 조합을 포함한다. 비록 "제1", "제2" 등이 다양한 구성요소들을 서술하기 위해서 사용되나, 이들 구성요소들은 이들 용어에 의해 제한되지 않음은 물론이다. 이들 용어들은 단지 하나의 구성요소를 다른 구성요소와 구별하기 위하여 사용하는 것이다. 따라서, 이하에서 언급되는 제1 구성요소는 본 발명의 기술적 사상 내에서 제2 구성요소일 수도 있음은 물론이다.

[0028] 다른 정의가 없다면, 본 명세서에서 사용되는 모든 용어(기술 및 과학적 용어를 포함)는 본 발명이 속하는 기술분야의 통상의 기술자에게 공통적으로 이해될 수 있는 의미로 사용될 수 있을 것이다. 또한, 일반적으로 사용되는 사전에 정의되어 있는 용어들은 명백하게 특별히 정의되어 있지 않는 한 이상적으로 또는 과도하게 해석되지 않는다.

[0029] 명세서에서 사용되는 "부" 또는 “모듈”이라는 용어는 소프트웨어, FPGA 또는 ASIC과 같은 하드웨어 구성요소를 의미하며, "부" 또는 “모듈”은 어떤 역할들을 수행한다. 그렇지만 "부" 또는 “모듈”은 소프트웨어 또는 하드웨어에 한정되는 의미는 아니다. "부" 또는 “모듈”은 어드레싱할 수 있는 저장 매체에 있도록 구성될 수도 있고 하나 또는 그 이상의 프로세서들을 재생시키도록 구성될 수도 있다. 따라서, 일 예로서 "부" 또는 “모듈”은 소프트웨어 구성요소들, 객체지향 소프트웨어 구성요소들, 클래스 구성요소들 및 태스크 구성요소들과 같은 구성요소들과, 프로세스들, 함수들, 속성들, 프로시저들, 서브루틴들, 프로그램 코드의 세그먼트들, 드라이버들, 펌웨어, 마이크로 코드, 회로, 데이터, 데이터베이스, 데이터 구조들, 테이블들, 어레이들 및 변수들을 포함한다. 구성요소들과 "부" 또는 “모듈”들 안에서 제공되는 기능은 더 작은 수의 구성요소들 및 "부" 또는 “모듈”들로 결합되거나 추가적인 구성요소들과 "부" 또는 “모듈”들로 더 분리될 수 있다.

[0030] 본 명세서에서, 컴퓨터는 적어도 하나의 프로세서를 포함하는 모든 종류의 하드웨어 장치를 의미하는 것이고, 실시 예에 따라 해당 하드웨어 장치에서 동작하는 소프트웨어적 구성도 포괄하는 의미로서 이해될 수 있다. 예를 들어, 컴퓨터는 스마트폰, 태블릿 PC, 데스크톱, 노트북 및 각 장치에서 구동되는 사용자 클라이언트 및 애플리케이션을 모두 포함하는 의미로서 이해될 수 있으며, 또한 이에 제한되는 것은 아니다.

[0031] 이하, 첨부된 도면을 참조하여 본 발명의 실시예를 상세하게 설명한다.

[0032] 본 명세서에서 설명되는 각 단계들은 컴퓨터에 의하여 수행되는 것으로 설명되나, 각 단계의 주체는 이에 제한되는 것은 아니며, 실시 예에 따라 각 단계들의 적어도 일부가 서로 다른 장치에서 수행될 수도 있다.

[0033] 일 예로, 본 명세서에서 언급되는 컴퓨터는 자동차에 설치되는, 적어도 하나의 프로세서를 포함하는 전장 시스템을 의미할 수 있으나, 이에 제한되는 것은 아니다.

[0034] 도 1은 일 실시 예에 따른 ECU의 CAN 에러 관리 방법을 도시한 도면이다.

[0035] 도 1을 참조하면, CAN 표준의 에러관리 기법이 도시되어 있다.

[0036] 일 실시 예에서, CAN의 에러관리를 위해 ECU에는 TEC(Transmission Error Counter) 및 REC (Receive Error Counter)가 존재할 수 있다.

[0037] 일 실시 예에서, ECU는 세가지의 Error 상태가 존재할 수 있으며, 예를 들어 Error Active, Error Passive 및 Bus-Off를 포함할 수 있다.

[0038] 일 실시 예에서, 특정 ECU가 Bus-Off상태가 되면, 해당 ECU는 일정시간동안 CAN 통신에 참여할 수 없게 된다.

[0039] 일 예로, 각 Error 상태 간의 이동(상태변환) 조건은 도 1에 도시된 바와 같을 수 있다.

[0040] 도 2는 일반적인 CAN-bus에 메시지 플러딩 공격이 발생한 일 예를 도시한 도면이다.

[0041] 도 2를 참조하면, 차량 내부의 전장시스템 및 그 네트워크가 도시되어 있다. 시스템에는 복수의 ECU가 포함되며, 각각의 ECU를 연결하는 CAN 네트워크가 존재한다.

[0042] 일 실시 예에서, 복수의 ECU 중 하나를 통한 공격이 수행되는 경우(예를 들어, 메시지 플러딩 공격으로 대표되는 도스(DoS) 공격이 자동차 내부 네트워크에서 발생할 경우), 도 2에 도시된 바와 같이 해당 공격 노드로부터 발생하는 F(메시지 플러딩 공격에 활용되는 CAN 메시지)메시지에 의해 다른 노드들(일반 ECU들)의 통신 장애가 발생하게 된다.

[0043] 일 실시 예에서, 메시지 플러딩 공격이 탐지되는 경우 동시에 에러 메시지를 통해 공격메시지를 폐기할 수 있다. 하지만, 공격 메시지 폐기의 영향으로 내부 네트워크의 패킷 스케쥴이 변경되는 문제가 발생할 수 있다.

[0044] 개시된 실시 예에 따르면, 차량 내부 네트워크에 메시지 플러딩 공격이 일어났을 경우, 공격 노드를 특정 시간 동안 차량 내부 네트워크에서 배제시킬 수 있다.

[0045] 이와 동시에, 개시된 실시 예에 따른 방법이 악용 혹은 오용되어 정상적인 노드들의 통신이 방해되는 현상이 없도록(e.g., 정상적인 ECU들이 CAN 네트워크에 배제되지 않도록) 하여야 하며, 이는 이하에서 도면을 참조하여 구체적으로 설명한다.

[0046] 도 3은 개시된 실시 예에 따른 프레임워크가 적용된 CAN-bus에 메시지 플러딩 공격이 발생한 일 예를 도시한 도면이다.

[0047] 도 3을 참조하면, 개시된 실시 예에 따른 프레임워크에 의하여 메시지 플러딩 공격을 발생시킨 공격 노드는 지속적으로 에러프레임(E)을 발생시키게 되며, 최종적으로 Bus-off상태가 된다.

[0048] Bus-off상태의 공격노드가 다시 CAN 통신에 참여하려면 일정시간이 지나야 하므로, 해당 시간동안 일반 ECU들은 메시지 플러딩 공격의 영향없이 CAN 메시지를 전송할 수 있다. 또한 에러프레임으로 인해 발생한 일반 메시지들의 지연시간을 예측함으로써, 개시된 실시 예에 따른 방법에 의해 일반 메시지가 잘못 폐기되는 것을 막을 수 있다.

[0049] 상술한 방법을 수행하는 구체적인 방법에 대해서는 도면을 참조하여 후술한다.

[0050] 도 4는 일 실시 예에 따라 CAN 네트워크에 대한 메시지플러딩 공격을 완화하는 방법을 도시한 흐름도이다.

[0051] 단계 S110에서, 컴퓨터는 일반 CAN 메시지들의 전송 스케쥴에 기반하여, 일반 CAN 메시지와 메시지 플러딩 공격에 이용된 CAN 메시지를 식별하는, 공격메시지 식별단계를 수행한다.

[0052] 단계 S120에서, 컴퓨터는 공격 노드가 메시지 플러딩 공격을 위해 다수의 CAN 메시지를 포함하는 공격 메시지를 전송할 때, 도미넌트(Dominant) 비트를 생성하여 상기 공격 노드에 CAN 에러 프레임 생성을 유도하는, 도미넌트 비트 생성단계를 수행한다.

[0053] 도미넌트 비트 전송은 Recessive 비트 전송에 비해 우선순위가 높으므로, 메시지 플러딩 공격 발생시 도미넌트 비트 생성 모듈이 지속적으로 도미넌트 비트들을 생성하게 되면, 공격 노드는 메시지 전송 과정에 에러가 발생하게 된다. 즉, 공격노드는 에러프레임을 발생시키면서 TEC를 증가시킨다. 공격노드의 TEC가 증가하여 255보다 커지면, 공격노드는 Bus-off 상태가 되어, 일정시간 CAN 통신에 참여할 수 없게 된다.

[0054] 단계 S130에서, 컴퓨터는 상기 도미넌트 비트로 인해 상기 공격 노드가 공격 메시지 전송을 실패하는 경우, 상기 공격 메시지를 재전송하기까지의 시간을 예측하는, 공격 메시지 재전송 시간($T_{retrans}$) 예측단계를 수행한다.

[0055] 상기 공격 메시지 재전송 시간 예측단계에서, 컴퓨터는 CAN의 에러 관리 절차를 획득하는 단계 및 상기 획득된

에러 관리 절차에 기초하여 상기 공격 메시지의 재전송 시간을 예측하되, 상기 도미넌트 비트로 인해 상기 공격 노드가 에러 프레임을 전송한 시점 및 상기 공격노드의 TEC(Transmission Error Counter)에 기초하여 상기 공격 메시지의 재전송 시간을 예측하는, 단계를 수행한다.

[0056] 예를 들어, CAN에서는 에러프레임으로 인해 CAN 메시지 전송 실패가 발생할 경우, 해당 메시지를 재전송하도록 규정하고 있다. 따라서 메시지 전송이 실패된 메시지는 에러 프레임 전송 후, Interframe space(3비트 전송시간) 후에 메시지 재전송을 시도한다. 하지만 공격노드의 TEC가 127을 넘게 되면, Suspend Transmission Time(8비트 전송시간)을 추가적으로 기다린 후, 메시지 재전송을 시도하게 된다.

[0057] 따라서, 컴퓨터는 공격 노드가 에러 프레임을 전송한 시점 및 상기 공격노드의 TEC에 기초하여 상기 공격 메시지의 재전송 시간을 예측할 수 있게 된다.

[0058] 단계 S140에서, 컴퓨터는 공격 메시지 및 에러 프레임으로 인하여 일반 CAN 메시지의 전송지연 발생시 상기 일반 CAN 메시지의 전송지연을 예측하는, 일반 CAN 메시지 전송지연 예측단계를 수행한다.

[0059] 단계 S130 및 단계 S140은 함께 수행될 수도 있으나, 둘 중 하나가 선택적으로 수행될 수도 있으며, 수행순서 및 선택방법은 제한되지 않는다.

[0060] 상기 일반 CAN 메시지 전송지연 예측단계에서, 컴퓨터는 상기 일반 CAN 메시지의 전송지연의 최악의 시간(WT_{delay})을 예측할 수 있다.

[0061] 상기 최악의 시간을 예측하는 단계에서, 컴퓨터는 아래 수학식에 기초하여 상기 최악의 시간을 예측할 수 있다.

[0062] $$WT_{delay} = T_{attack} + T_{retrans} + T_{error}$$

[0063] 상기 수학식에서, T_{attack}은 상기 공격 메시지로 인한 지연시간을 의미하고, 상기 T_{error}는 상기 에러 프레임으로 인한 지연시간을 의미한다.

[0064] 일 실시 예에서, CAN 메시지의 전송지연시간의 최악의 경우를 고려하여, 일반 CAN 메시지의 발생 가능한 전송지연을 예측할 수 있다. 일반 CAN 메시지의 전송지연시간이 최악의 경우인 보다 작을 경우, 해당 정보를 일반메시지로 분류함으로써 메시지 식별 에러를 방지할 수 있다.

[0065] 도 5는 다른 일 실시 예에 따른 메시지플러딩 공격 완화방법을 도시한 흐름도이다.

[0066] 단계 S210에서, 컴퓨터는 상기 공격 노드로부터 발생하는 공격 메시지의 패턴을 획득할 수 있다.

[0067] 단계 S220에서, 컴퓨터는 데이터베이스로부터 상기 패턴에 대응하는 공격수단에 대한 정보를 획득할 수 있다.

[0068] 단계 S230에서, 컴퓨터는 상기 획득된 정보로부터 하나 이상의 파라미터를 추출할 수 있다.

[0069] 단계 S240에서, 컴퓨터는 상기 추출된 파라미터에 기반하여 상기 공격 메시지 재전송 시간 및 상기 일반 CAN 메시지 전송지연 예측결과를 조정할 수 있다.

[0070] 또한, 다른 일 실시 예에서, 공격 노드가 Bus off 되는 경우, 컴퓨터는 상기 공격 노드에 대응하는 예비 노드를 상기 CAN 네트워크에 연결할 수 있다. 또한, 컴퓨터는 상기 예비 노드가 상기 공격 노드의 역할을 수행하도록 제어할 수 있다.

[0071] 일 실시 예에서, 시스템은 하나 이상의 ECU 및 하나 이상의 ECU를 연결하는 CAN 네트워크를 포함할 수 있다.

[0072] 일 실시 예에서, 상기 하나 이상의 ECU중 하나인 제1 ECU에 대하여, 상기 시스템은, 상기 제1 ECU와 동일한 기능을 수행하는 제2 ECU를 더 포함할 수 있다.

[0073] 일 실시 예에서, 상기 제1 ECU가 공격 노드가 되어 Bus off 되는 경우, 상기 제2 ECU가 상기 CAN에 연결되어 상기 제1 ECU의 역할을 수행할 수 있다.

[0074] 또한, 상기 하나 이상의 ECU중 하나인 제3 ECU에 대하여, 상기 제3 ECU를 제외한 상기 하나 이상의 ECU는, 상기 제3 ECU가 수행하는 기능 각각에 대응하는 적어도 하나의 모듈을 포함할 수 있다.

[0075] 일 예로, 제3 ECU에 포함된 모든 모듈들과 동일한 역할을 수행하는 모듈들이 각각 나머지 ECU들에 포함될 수 있다.

[0076] 일 실시 예에서, 상기 제3 ECU가 공격 노드가 되어 Bus off 되는 경우, 상기 제3 ECU를 제외한 상기 하나 이상

의 ECU 각각에 포함된 모듈들이 상기 제3 ECU의 기능을 수행하도록 할 수 있다.

[0077] 상술한 바와 같이, 중요한 역할을 수행하는 ECU가 공격으로 인해 Bus off되는 경우 전체 시스템의 구동에 문제가 생길 수 있으므로, 이에 대비하기 위한 예비수단이 마련될 수 있다.

[0078] 이러한 예비수단은 각각의 ECU가 Bus off되었을 때 이용되며, 실시 예에 따라 각각의 ECU가 공격 노드가 되어 에러 프레임을 발생시켜 Bus off가 되는 경우, 시스템은 해당 공격 노드의 공격 메시지 재전송 시간을 예측할 수 있다. 이때, 시스템은 해당 재전송 시간까지의 간격 동안 해당 공격 노드의 Bus off 상태를 해제할 수 있다. 이 경우, 예비수단은 해당 공격 노드의 Bus off 상태가 해제되는 경우 Bus off 되고, 공격메시지 재전송 시간으로부터 소정의 범위 내의 시간에 해당 공격 노드가 다시 Bus off 되는 경우 다시 CAN에 연결될 수도 있다.

[0079] 일 실시 예에서, 시스템은 예측된 공격 메시지 재전송 시간으로부터 소정의 범위 내의 시간이 되는 경우, 해당 공격 노드로부터 발생하는 공격 메시지를 탐지하되, 공격 메시지를 탐지하는 기준을 평소보다 낮출 수 있다. 즉, 공격 메시지인 것으로 판단하는 기준을 평소보다 낮춤으로써, 공격 메시지인 것으로 판단되는 요소가 평소보다 적어도 즉시 공격메시지를 발생하는 것으로 인식하여 해당 공격노드가 Bus off 되도록 할 수 있다.

[0080] 일 실시 예에서, 시스템은 공격 노드로부터 발생되는 메시지의 패턴을 분석할 수 있다.

[0081] 예를 들어, 시스템은 공격 노드에서 발생되는 공격 메시지의 간격, 시간 및 이를 포함하는 패턴을 분석할 수 있고, 기 저장된 데이터베이스에 근거하여 패턴에 매칭되는 공격수단을 획득할 수 있다.

[0082] 예를 들어, 시스템은 패턴에 매칭되는 공격수단을 획득함으로써, 해당 공격수단에 대응하는 하나 이상의 파라미터를 획득할 수 있다.

[0083] 시스템은 획득된 파라미터에 기반하여, 상술한 모듈들에 의하여 계산되는 공격 메시지 재전송 시간 및 일반 CAN 메시지 전송지연 예측결과를 조정할 수 있다.

[0084] 도 6은 일 실시 예에 따른 장치의 구성도이다.

[0085] 프로세서(102)는 하나 이상의 코어(core, 미도시) 및 그래픽 처리부(미도시) 및/또는 다른 구성 요소와 신호를 송수신하는 연결 통로(예를 들어, 버스(bus) 등)를 포함할 수 있다.

[0086] 일 실시예에 따른 프로세서(102)는 메모리(104)에 저장된 하나 이상의 인스트럭션을 실행함으로써, 도 1 내지 도 5와 관련하여 설명된 방법을 수행한다.

[0087] 예를 들어, 프로세서(102)는 메모리에 저장된 하나 이상의 인스트럭션을 실행함으로써 개시된 실시 예에 따른 메시지플러딩 공격 완화방법을 수행할 수 있다.

[0088] 한편, 프로세서(102)는 프로세서(102) 내부에서 처리되는 신호(또는, 데이터)를 일시적 및/또는 영구적으로 저장하는 램(RAM: Random Access Memory, 미도시) 및 롬(ROM: Read-Only Memory, 미도시)을 더 포함할 수 있다. 또한, 프로세서(102)는 그래픽 처리부, 램 및 롬 중 적어도 하나를 포함하는 시스템온칩(SoC: system on chip) 형태로 구현될 수 있다.

[0089] 메모리(104)에는 프로세서(102)의 처리 및 제어를 위한 프로그램들(하나 이상의 인스트럭션들)을 저장할 수 있다. 메모리(104)에 저장된 프로그램들은 기능에 따라 복수 개의 모듈들로 구분될 수 있다.

[0090] 본 발명의 실시예와 관련하여 설명된 방법 또는 알고리즘의 단계들은 하드웨어로 직접 구현되거나, 하드웨어에 의해 실행되는 소프트웨어 모듈로 구현되거나, 또는 이들의 결합에 의해 구현될 수 있다. 소프트웨어 모듈은 RAM(Random Access Memory), ROM(Read Only Memory), EPROM(Erasable Programmable ROM), EEPROM(Electrically Erasable Programmable ROM), 플래시 메모리(Flash Memory), 하드 디스크, 착탈형 디스크, CD-ROM, 또는 본 발명이 속하는 기술 분야에서 잘 알려진 임의의 형태의 컴퓨터 판독가능 기록매체에 상주할 수도 있다.

[0091] 본 발명의 구성 요소들은 하드웨어인 컴퓨터와 결합되어 실행되기 위해 프로그램(또는 애플리케이션)으로 구현되어 매체에 저장될 수 있다. 본 발명의 구성 요소들은 소프트웨어 프로그래밍 또는 소프트웨어 요소들로 실행될 수 있으며, 이와 유사하게, 실시 예는 데이터 구조, 프로세스들, 루틴들 또는 다른 프로그래밍 구성들의 조합으로 구현되는 다양한 알고리즘을 포함하여, C, C++, 자바(Java), 어셈블러(assembler) 등과 같은 프로그래밍 또는 스크립팅 언어로 구현될 수 있다. 기능적인 측면들은 하나 이상의 프로세서들에서 실행되는 알고리즘으로 구현될 수 있다.

[0092] 이상, 첨부된 도면을 참조로 하여 본 발명의 실시예를 설명하였지만, 본 발명이 속하는 기술분야의 통상의 기술자는 본 발명이 그 기술적 사상이나 필수적인 특징을 변경하지 않고서 다른 구체적인 형태로 실시될 수 있다는 것을 이해할 수 있을 것이다. 그러므로, 이상에서 기술한 실시예들은 모든 면에서 예시적인 것이며, 제한적이 아닌 것으로 이해해야만 한다.

부호의 설명

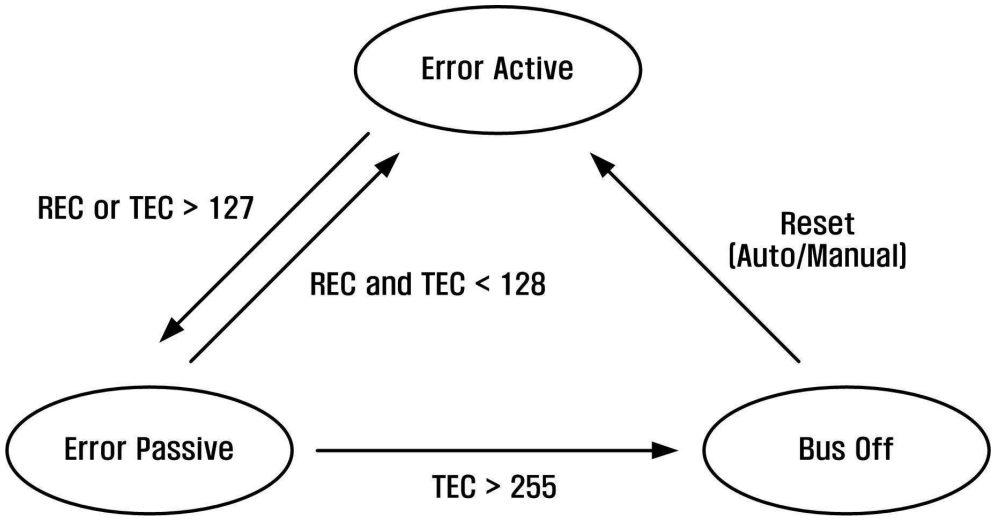
[0093] 100 : 장치

102 : 프로세서

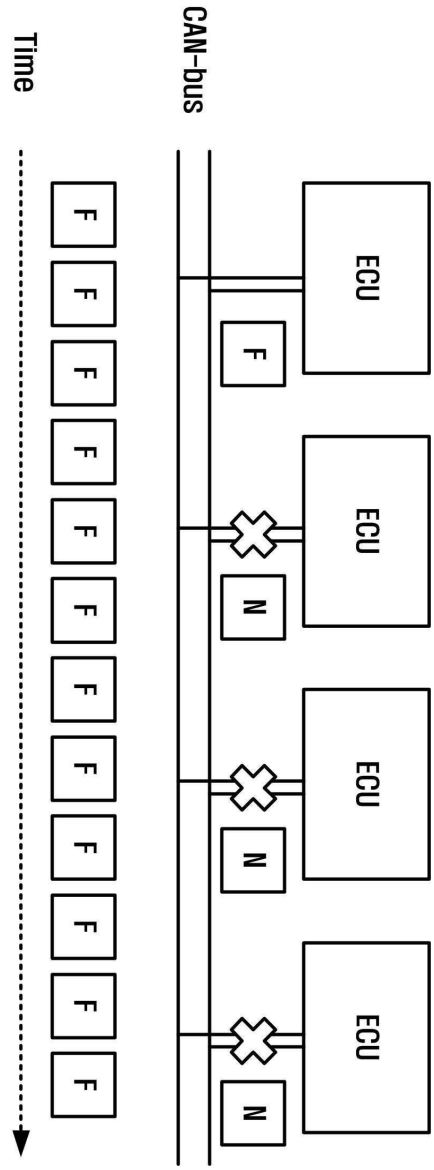
104 : 메모리

도면

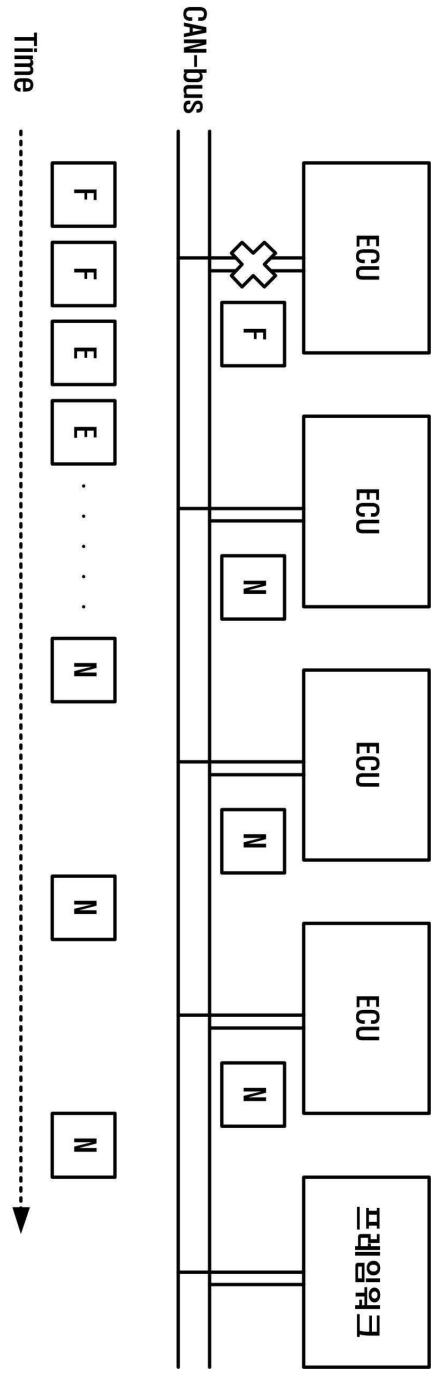
도면1



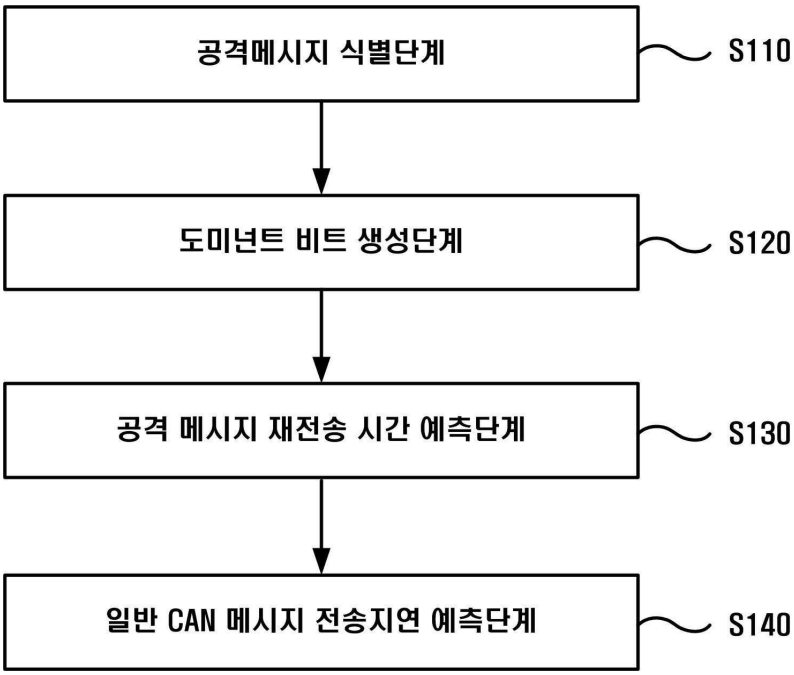
도면2



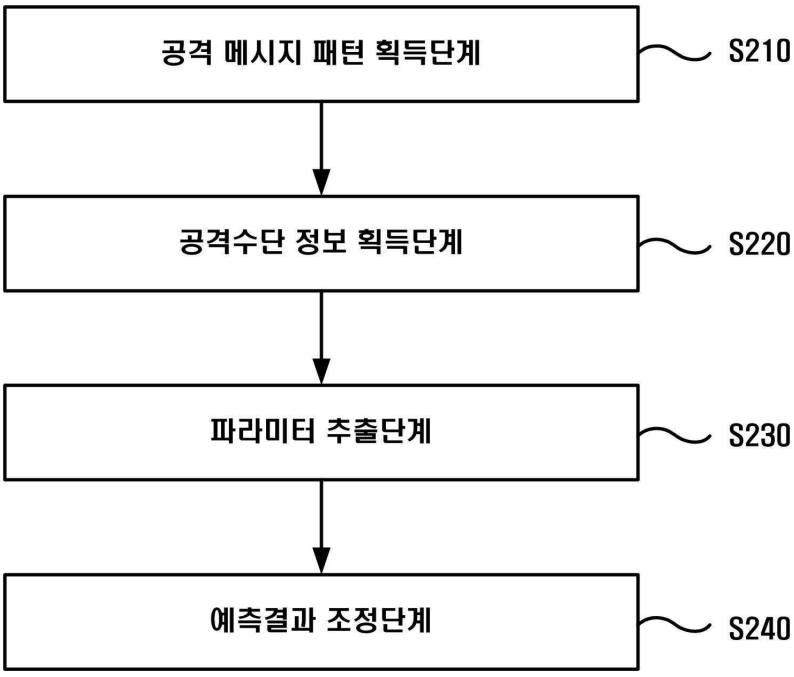
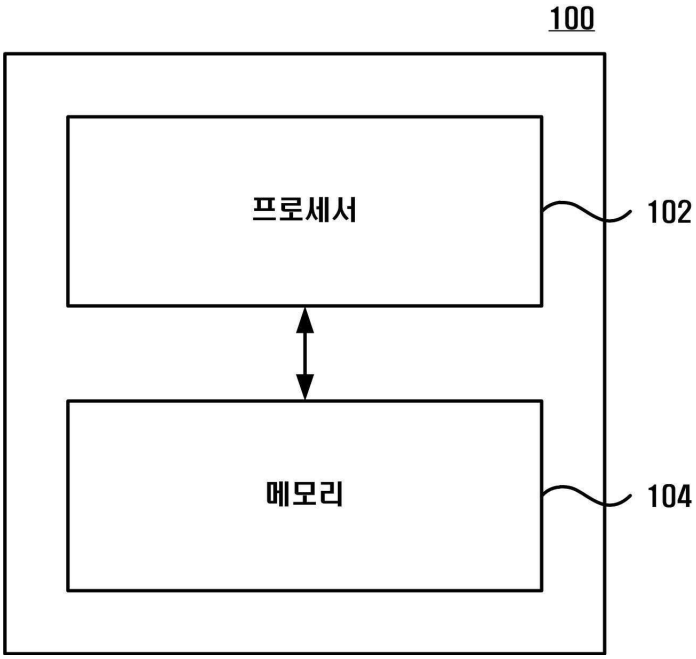
도면3



도면4


공격메시지 식별단계
S110
도미넌트 비트 생성단계
S120
공격 메시지 재전송 시간 예측단계
S130
일반 CAN 메시지 전송지연 예측단계
S140

도면5


공격 메시지 패턴 획득단계
S210
공격수단 정보 획득단계
S220
파라미터 추출단계
S230
예측결과 조정단계
S240

도면6