Journal of the Korea Convergence Society
Vol. 8. No. 3, pp. 23-30, 2017 https://doi.org/10.15207/JKCS.2017.8.3.023

OpenADR 2.0b 페이로드 트랜잭션 분석 모델 연구

최민영[1], 이준경[1*], 이경학[2]
[1](주)나온웍스, [2]광운대학교 산학협력단

A Transaction Analysis Model for OpenADR 2.0b Payload

Min-Young Choi[1], June-Kyoung Lee[1*], Kyoung-Hak Lee[2]
[1]NAONWORKS Co.,Ltd.
[2]IACF Kwangwoon University

요 약 유동적인 전기 소비량과 소비자와 공급자가 실시간으로 변하는 시장 상황에 대응하기 위해 수요 반응에 기술이 있다. OpenADR은 이러한 수요 반응 서비스를 제공하는 국제적인 표준 통신 프로토콜이다. OpenADR은 어플리케이션 계층에서 페이로드라는 XML 메시지를 사용하며, 본 프로토콜이 제공하는 수요 반응 서비스는 페이로드에 논리적인 요청, 응답을 사용해 제공한다. 이를 위해, OpenADR은 트랜잭션에 식별자로 'requestID'라는 XML 엘리먼트를 정의한다. OpenADR 2.0b 프로파일 사양 문서에서 트랜잭션 식별자에 설명은 있지만, 이해하기에 충분하지 않다. 모호한 트랜잭션에 설명은 OpenADR에 VEN과 VTN 구현에 취약점을 만들고, 서로 다른 프로토콜 간 변환에 혼란을 줄 가능성이 있다. 따라서 본 논문에서는 OpenADR 2.0b에 페이로드 트랜잭션 모호성을 해결하는 정의와 페이로드 트랜잭션을 분석하는 모델을 제시하고, 본 논문에서 제시한 분석 모델을 사용해 실제 트랜잭션 취약점을 탐지하는 사례를 보인다.

• **주제어** : 스마트 그리드 보안, 융합 보안, 수요 반응, 개방형 수요 반응, 트랜잭션 분석

Abstract OpenADR is a national standard communication protocol of demand response service. OpenADR uses XML message, called a payload, to support logical transaction of demand response service. For this purpose, OpenADR defines a XML element as transaction identifier which is called requestID. Unfortunately, OpenADR 2.0b profile specification describes some information about requestID but, it is not enough for understanding properly. Ambiguous definition of payload transaction makes vulnerabilities of implementing VEN & VTN and confuses mapping OpenADR 2.0b protocol into other smart grid protocols. Therefore, this paper redefines payload transaction to solve an ambiguous information of OpenADR 2.0b profile specification, proposes a model of analyzing payload transaction, and shows how to detect a payload transaction vulnerability in real-world

• **Key Words** : Smart Grid Security, Convergence Security, Demand Response, OpenADR 2.0b, Transaction Analysis

*Corresponding Author : 이준경(darkelan@naonworks.com)
Received February 7, 2017 Revised February 21, 2017
Accepted March 20, 2017 Published March 28, 2017

1. 서론

스마트 그리드란 전기의 발전, 송전, 배전, 과정에 정보통신기술(information & communication technology, ICT)을 접목하여 공급자와 소비자가 서로 상호 작용함으로써 전력망의 지능화 및 고도화를 추구하고 고품질의 전력 서비스를 제공하여 에너지 이용 효율을 극대화하는 차세대 전력망으로써 융합 보안 기술의 접목이 반드시 필요한 분야이다[1,2,3].

스마트 그리드 망에서 수요 반응(demand response, DR)은 시간 경과에 따른 유동적인 전기 가격 변화에 맞추어 소비자에 전기 사용량 변화, 시장 가격에 변화, 관리 시스템에 안정성이 위험에 처할 때, 전기 사용량을 조절하기 위해 고안한 기술이라 정의한다[4]. 수요 반응에 안정성과 신뢰성을 위해서는 표준화된 통신 방식과 데이터 형식이 필요하며. 시간대별 차등요금제와 같은 다양한 정책적 수단들이 수요 반응에 활성화를 위해 개발되고 있다[5,6,7,8].

개방형 자동 수요 반응(open automated demand response, OpenADR)은 지능형 수요 반응에 사용하는 국제 표준 통신 프로토콜이며, 어플리케이션 계층으로 설계되어 상호 운영적이다. 그렇게 함으로써 시스템에 종속적이지 않다. 또한 페이로드(payload)라 하는 XML 전송 메시지를 사용하며, VEN(virtual end node)과 VTN(virtual top node)에 수요 반응 서비스(2.0a/2.0b) 통신에 사용한다[9,10,11,12].

어플리케이션 계층에서 페이로드는 요청과 응답에 논리적 트랜잭션으로 서비스를 수행하고, 이를 위해 'requestID'라는 트랜잭션 식별자를 정의한다[9,10,13]. OpenADR은 자체적인 보안 정책을 가지고 있지만[10,14], OpenADR 프로파일 사양 문서에 모호하게 적힌 설명이 페이로드 트랜잭션과 관련한 OpenADR 오픈소스 취약점을 만들고 있다[10,15,16,17]. SEP 2.0, MIRABEL과 같은 프로토콜과 OpenADR 2.0b에 변환(mapping)이 이루어지고 있는 상황에서[18,19], 트랜잭션 식별자에 모호성을 해결하는 것이 또 다른 취약점을 만들지 않는 열쇠이다.

본 논문에서는 더 이상 지원하지 않는 OpenADR 2.0a는 제외하며[10], OpenADR 2.0b 페이로드 트랜잭션에 모호성을 해결하는 정의와 분석 모델을 제시하고, 해당 모델을 사용하여 트랜잭션 취약점을 탐지하는 방법을 제시한다.

2. 페이로드 트랜잭션

2.1 트랜잭션 정의

트랜잭션에 모호성을 없애기 위해서는 페이로드에 트랜잭션을 정확히 정의해야 한다. OpenADR 2.0b의 요청 페이로드와 응답 페이로드에 논리적인 짝을 맞춰주는 식별자를 '트랜잭션 아이디'라 정의한다. OpenADR 2.0b는 'requestID'를 페이로드에 트랜잭션 아이디라 하며 특징은 다음과 같다[10,13].

- EiEvent 서비스에서 requestID는 요청 페이로드와 페이로드가 포함하는 이벤트를 유일하게 구분한다.
- VEN은 VTN이 이벤트 요청시 생성한 requestID에 유일성을 의심하지 않는다. VTN이 원하는 방식으로 생성할 수 있다.
- EiOpt, EiReport 서비스에 요청 페이로드가 requestID를 명시하면, 반드시 응답 페이로드에 해당 requestID를 인용해야 한다.

위에 서술한 특징은 트랜잭션의 독립적인 정의 없이 서비스 별로 산개된 설명이다. 하지만 위에 언급한 대로 트랜잭션 아이디가 논리적으로 페이로드를 연결한다면 트랜잭션에 정의는 다음과 같아야 한다.

1) 페이로드에 트랜잭션 아이디로 requestID를 사용한다.
2) 요청 페이로드에 requestID는 해당 페이로드에 요청을 유일하게 구분하며, 이에 논리적인 응답 페이로드는 요청 페이로드가 생성한 requestID를 응답에 반드시 인용해야 한다. 이를 논리적 트랜잭션이라 한다.
3) 요청 페이로드가 requestID를 생성하지 않으면, 이에 논리적인 응답 페이로드는 requestID를 인용하지 않고, 두 페이로드는 논리적 트랜잭션이 아니다.

새로운 트랜잭션 정의는 기존에 서술한 트랜잭션에 특징을 포함한다. 또한 서비스에 상관없이 일반적인 페이로드를 사용하여 기술하였다. 이렇게 함으로써 트랜잭션 아이디를 생성하는 모든 페이로드는 논리적인 트랜잭션 생성을 보장받는다.

OpenADR 2.0b에서는 'oadrPayload', 'oadrSignedObject'

에 XML 랩퍼(wrapper) 엘리먼트를 제외한 상위 엘리먼트를 서비스 오퍼레이션(service operation)이라 정의한다[13]. 이를 참고하여 트랜잭션 아이디를 구조적으로 설명하면 다음과 같다.

- 요청 페이로드에 트랜잭션 아이디는 해당 페이로드에 서비스 오퍼레이션을 정의하는 엘리먼트 아래에 정의한다. 이후 해당 아이디를 '요청 트랜잭션 아이디'라 칭한다.
- 응답 페이로드에 트랜잭션 아이디는 해당 페이로드에 서비스 오퍼레이션을 정의하는 엘리먼트 아래에 'eiResponse' 엘리먼트를 정의하고, eiResponse 엘리먼트 아래에 정의한다. 이후 해당 아이디를 '응답 트랜잭션 아이디'라 칭한다.

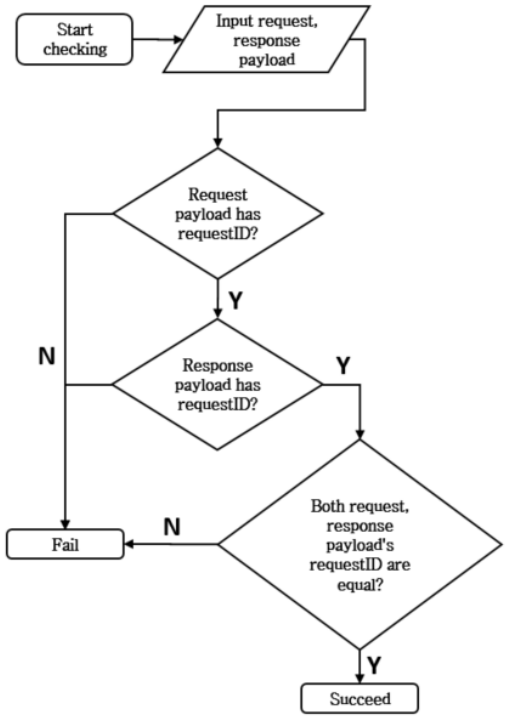
2.2 트랜잭션 실패

트랜잭션에 성공과 실패를 판단하는 것은 트랜잭션 아이디를 올바르게 사용했는지를 검사하는 것이다. 트랜잭션 판별에 기본 흐름은 [Fig. 1]과 같다. '2.1 트랜잭션 정의'에서 서술한 대로, 요청, 응답 페이로드에 트랜잭션 아이디가 존재하면 해당 아이디의 문자열에 상관없이 항상 일치해야 한다. 일반적으로 요청 페이로드에 대한 응답 페이로드는 이미 정해져 있지만[10,13], 모든 페이로드가 트랜잭션이 성립하는 것은 아니다. 왜냐하면 트랜잭션 아이디를 정의하지 않는 페이로드가 존재하기 때문이다. 또한 비동기적인 전송 계층 통신에서 논리적 트랜잭션이 발생 할 수 있다. 이러한 페이로드는 정상적인 서비스 시나리오 범주이기에 트랜잭션 실패라 할 수 없다. 자세한 사항은 '2.3 트랜잭션 예외'에서 서술한다.

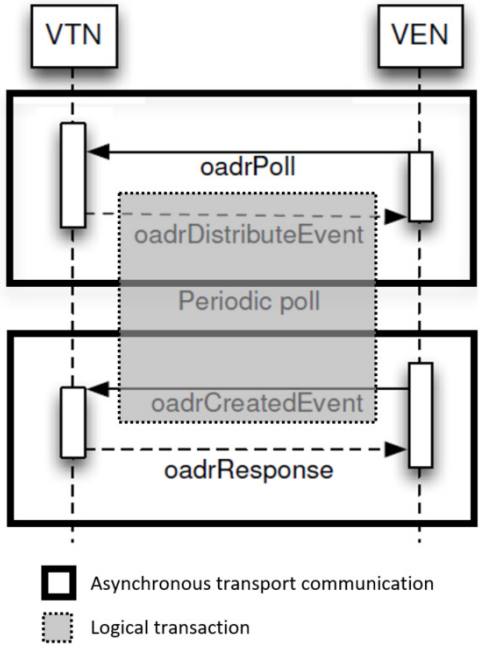
2.3 트랜잭션 예외

트랜잭션 아이디를 정의하지 않거나 사용하지 않는 페이로드와 비동기적으로 전송 계층에 영향을 받는 트랜잭션에 경우는, 직관적으로 [Fig. 1]에 알고리즘을 사용할 수 없다. 예외 상황은 다음에 두 가지다.

1) 논리적 트랜잭션에 PULL 모드 확장
2) 논리적 트랜잭션이 필요 없는 전송 계층 통신



[Fig. 1] Algorithm for transaction check



[Fig. 2] EiEvent PULL mode scenario[10]

또한, OpenADR 2.0b는 VEN과 VTN에 통신 방법으로 다음에 두 가지를 사용한다[10].

- PUSH mode - VEN과 VTN이 서비스 시나리오에 따라 서로 통신할 수 있다.
- PULL mode - VTN은 VEN이 폴링(polling)에 사용하는 oadrPoll 페이로드를 받아야지만 서비스를 시작 할 수 있다.

OpenADR 2.0b가 제공하는 PULL 모드에서는, PUSH 모드에서 사용하는 논리적 트랜잭션 시나리오를 사용하려고 oadrPoll 서비스를 정의한다. oadrPoll 서비스를 사용해 VEN은 폴링 메커니즘을 실행한다. 폴링 메커니즘은 VTN이 수행할 서비스가 있으면, 최소 두 번에 비동기적인 전송 계층 요청, 응답 통신을 수행한다. 이 때, 아래에 서술한 페이로드가 트랜잭션 예외 1)을 만든다.

- oadrPoll
- oadrResponse

[Fig. 2]는 트랜잭션 예외 1)에 시나리오다. oadrPoll 페이로드는 트랜잭션 아이디를 정의하지 않으며, oadrResponse 페이로드는 트랜잭션 아이디를 정의하지만 사용하지 않는다. 왜냐하면 논리적 트랜잭션 때문이다. oadrCreatedEvent 페이로드는 oadrDistributeEvent 페이로드에 논리적 응답 페이로드다. 또한 oadrCreatedEvent 페이로드는 응답 트랜잭션 아이디만 사용하고, 요청 트랜잭션 아이디는 정의하지 않는다. 결국 어플리케이션 계층에서 oadrCreatedEvent 페이로드에 대한 응답으로 사용할 페이로드는 응답 트랜잭션 아이디로 사용할 아이디가 없다. 따라서 oadrCreatedEvent 페이로드에 응답으로 사용한 oadrResponse 페이로드는 비록 응답 트랜잭션 아이디를 정의하지만 문맥상 사용할 수 없는 것이다. 중요한 점은 OpenADR 2.0b는 논리적 트랜잭션으로 정의하는 서비스 시나리오가 끝나고, 전송 계층 통신이 끝날 때까지, 종료한 트랜잭션 아이디를 재사용하지 않는다. [Fig. 2]에 'Asynchronous transport communication'이 이에 해당한다.

트랜잭션 예외 2)에 경우는 간단하다. 현재 OpenADR 2.0b에서 트랜잭션 아이디를 정의하지 않는 페이로드는 다음에 두 가지다.

- oadrRequestReregistration
- oadrPoll

위 두 가지 페이로드를 사용하는 시나리오가 한 가지 있다. VEN에 재등록 시나리오다. PULL 모드라면, VEN이 oadrPoll 페이로드로 폴링하여 VTN은 oadrRequestReregistration 페이로드로 응답을 한다. 이 경우 oadrPoll, oadrRequestReregistration 페이로드가 트랜잭션 아이디를 정의하지 않기 때문에 논리적 트랜잭션이 아니다. 논리적 트랜잭션은 비동기적으로 다음에 실행될 것이다. PUSH 모드라면, VTN이 VEN에게 바로 oadrRequestReregistration 페이로드로 요청한다. VEN은 전송 계층에 응답을 위해 oadrResponse 페이로드로 응답을 한다. 이 경우, 위에서 설명한 트랜잭션 예외 1)과 유사하다. oadrResponse 페이로드는 이 경우에도 문맥상 응답 트랜잭션 아이디를 사용하지 않아야 한다.

3. 페이로드 트랜잭션 분석 모델

3.1 페이로드 분석 타입 정의

본 논문에서는 페이로드 트랜잭션 분석을 위해 페이로드에 타입을 새로 정의한다. <Table 1>은 타입 별로 OpenADR 2.0b 서비스에 해당하는 페이로드와 페이로드에 방향성을 정리했다. 새로 정의할 타입은 다음에 네 가지다.

- Generator
- Responsor
- Moderator
- None

'Generator' 타입은 요청 트랜잭션 아이디만 생성하는 페이로드를 의미한다. 'Responsor' 타입은 응답 트랜잭션 아이디만 생성하는 페이로드를 의미한다. 'Moderator' 타입은 요청, 응답 트랜잭션 아이디를 생성할 수 있는 페이로드를 의미한다. 'None' 타입은 트랜잭션 아이디를 생성하지 않는 페이로드를 의미한다. 페이로드의 방향성은 다음에 세 가지를 정의한다.

- VEN→VTN
- VTN→VEN
- Both

〈Table 1〉 OpenADR 2.0b Payload transaction

Transaction Type	Service Type	Payload	Direction	Remarks
Generator	EiEvent	oadrRequestEvent	VEN→VTN	-
	EiReport	oadrCancelReport	Both	-
		oadrCreateReport	Both	-
		oadrRegisterReport	Both	-
		oadrUpdateReport	Both	-
	EiOpt	oadrCancelOpt	VEN→VTN	-
		oadrCreateOpt	VEN→VTN	-
	EiRegisterParty	oadrCancelPartyRegistration	Both	-
		oadrCreatePartyRegistration	VEN→VTN	-
		oadrQueryRegistration	VEN→VTN	-
	OadrPoll	-	-	NA
Responsor	EiEvent	oadrCreatedEvent	VEN→VTN	eventResponse
		oadrResponse	VTN→VEN	-
	EiReport	oadrCanceledReport	Both	-
		oadrCreatedReport	Both	-
		oadrRegisteredReport	Both	-
		oadrResponse	Both	-
	EiOpt	oadrCanceledOpt	VTN→VEN	-
		oadrCreatedOpt	VTN→VEN	-
	EiRegisterParty	oadrCanceledPartyRegistration	Both	-
		oadrCreatedPartyRegistration	VTN→VEN	-
		oadrResponse	Both	-
	OadrPoll	-	-	NA
Moderator	EiEvent	oadrDistributeEvent	VTN→VEN	-
	EiReport	oadrUpdatedReport	Both	oadrCancelReport
	EiOpt	-	-	NA
	EiRegisterParty	-	-	NA
	OadrPoll	-	-	NA
None	EiEvent	-	-	NA
	EiReport	-	-	NA
	EiOpt	-	-	NA
	EiRegisterParty	oadrRequestReregistration	VTN→VEN	-
	OadrPoll	oadrPoll	VEN→VTN	-

NA - Not available for transaction type

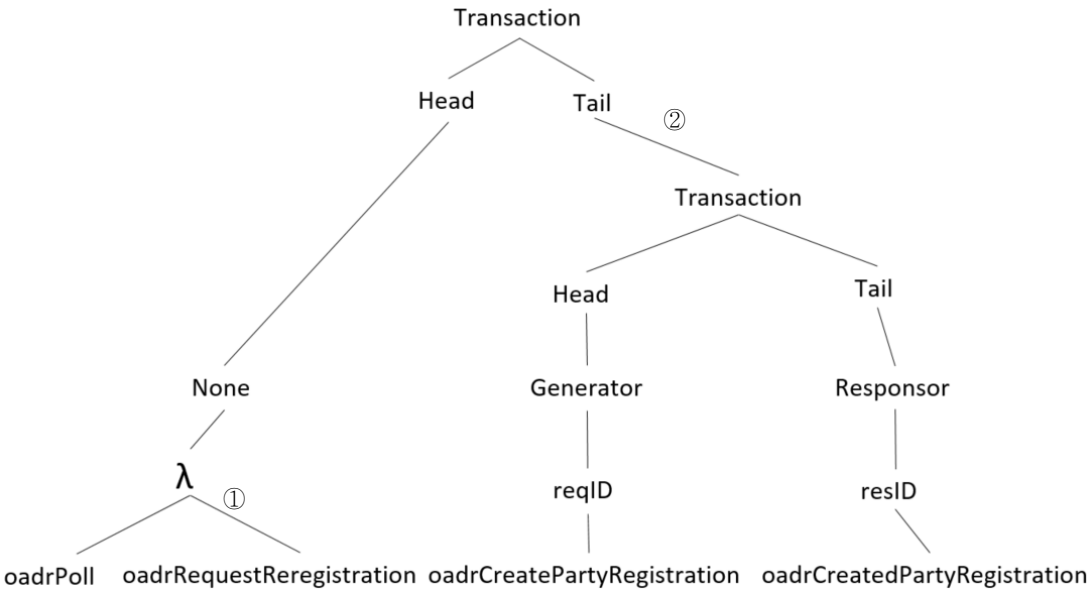
'VEN→VTN' 방향성은, 페이로드가 VEN에서 작성되어 VTN으로 전달함을 의미한다. 'VTN→VEN' 방향성은, 페이로드가 VTN에서 작성되어 VEN으로 전달함을 의미한다. 'Both' 방향성은, VEN→VTN, VTN→VEN을 동시에 가진다.

<Table 1>에 Remakrs가 작성된 페이로드가 두 가지 있다. 다른 페이로드와는 트랜잭션이 복잡한 페이로드다. oadrCreatedEvent 페이로드는 응답 트랜잭션 아이디를 정의하지만 실제로 응답할 이벤트가 있으면 응답 트랜잭션 아이디가 아니라, oadrCreatedEvent 페이로드에 하위 XML 엘리먼트인 eventResponse 엘리먼트에 requestID 엘리먼트를 응답 트랜잭션 아이디로 사용한다. oadrUpdatedReport 페이로드는 응답 트랜잭션 아이디를 사용하고, 하위에 oadrCancelReport 페이로드를 참조하여 요청 트랜잭션 아이디를 정의한다. 두 페이로드가 추가로 정의하는 트랜잭션 아이디를 복수로 정의할 수 있어, 올바른 트랜잭션을 종료하려면 모든 트랜잭션 아이디에 트랜잭션을 종료해야한다.

3.2 트랜잭션 확장과 생성 규칙 정의

OpenADR 2.0b에 페이로드 트랜잭션을 분석하기 위해서는 논리적 트랜잭션뿐만 아니라 트랜잭션 예외도 처리해야 한다. 이를 위해 본 논문에서는 트랜잭션을 확장하여 어플리케이션 계층이 사용하는 모든 페이로드에 트랜잭션을 분석하고자 한다.

본 논문은 페이로드 트랜잭션 모델이 사용하는 알고리즘으로 구문 패턴 인식(syntactic pattern recognition) 기법을 활용하고자[20,21], '3.1 페이로드 분석 타입 정의'에서 정의한 페이로드 타입을 사용하여 페이로드 트랜잭션 생성규칙을 CFG(context free grammar)로 표현하여



[Fig. 3] An example of payload transaction reduction for EiRegisterParty re-registering service

정의한다.

Transaction → Head Tail
Head → Generator | Moderator | None
Tail → Transaction | Responsor | None
Generator → reqID
Responsor → resID | λ
Moderator → reqID resID | reqID | resID
None → λ

위 생성규칙에 첫 글자가 대문자면 논터미널(nonterminal), 소문자면 터미널(terminal)이다. 람다(λ)에 의미는 트랜잭션 아이디가 없음을 의미하고, reqID는 요청 트랜잭션 아이디가 존재함을 의미하며, resID는 응답 트랜잭션 아이디가 존재함을 의미한다. 본 생성 규칙은 '2.1 페이로드 트랜잭션'에서 정의한 페이로드 트랜잭션과 트랜잭션에 예외를 포함한다. 어떤 페이로드가 위에 생성규칙을 사용해 시작 논터미널인 Transaction으로 감축하려면, <Table 1>을 사용한다. 즉, oadrPoll 페이로드는 언제나 트랜잭션 아이디가 없기 때문에, 람다이며 <Table 1>을 사용해 None으로 감축 할 수 있다.

3.3 트랜잭션 패턴탐지 기법과 적용

'3.2 트랜잭션 확장과 생성 규칙'에서 정의한 트랜잭션 생성규칙을 활용해 OpenADR 2.0b에 서비스 시나리오를 패턴으로 정의할 수 있다. 정의한 패턴은 페이로드 트랜잭션 탐지에 용도로 사용한다. 예를 들어 [15]에서 VEN 재등록 시나리오에 취약점을 보였는데, 이를 패턴탐지 기법을 적용해 [Fig. 3]과 같이 해결 할 수 있다. 재등록 시나리오에 사용한 각 페이로드는 해당하는 타입에 논터미널로 변환하며, 상향식으로 매 감축마다 해당하는 페이로드 타입에 트랜잭션을 검사하는 것이다. [Fig. 3]에 ①에 감축은 두 페이로드가 모두 람다로 감축된다. 이렇게 할 수 있는 이유는 두 페이로드가 모두 트랜잭션 아이디가 없는 None 타입이며, 전송 계층에서 요청과 응답으로 묶을 수 있기 때문이다. ②에 감축에서 트랜잭션에 취약점을 탐지 할 수 있다. ②에 감축에서 생성한 Transaction 논터미널의 감축 규칙이 활성화된 Head 논터미널에 패턴이 oadrPoll, oadrRequestReregistration 페이로드에 감축으로 생성한 것이 맞는지 확인하는 것이다. 이와 같은 Head 논터미널이 존재 한다면 감축에 성공하여 정상적인 시나리오 패턴을 탐지 할 수 있다. 그렇지 않은 경우, ②에서 감축을 위해 생성한 Transaction 논터미널이 oadrCreatePartyRegistration, oadrCreated

-PartyRegistration 페이로드로 생성한 패턴이 아니라면, 더 이상에 감축은 필요 없지만 이전 Head 논터미널이 감축을 기다리고 있으므로 상기에 Transaction 논터미널이 가진 패턴은 정상 시나리오 패턴이 아닌 것을 알 수 있다.

4. 결론

본 논문에서는, 기존 OpenADR 2.0b 프로파일 사양[10]에서 모호했던 트랜잭션 설명을 정립하고, 트랜잭션 아이디를 가지지 않는 페이로드를 트랜잭션으로 확장하였다. 또한, 본 논문에서 제시한 OpenADR 2.0b 페이로드 트랜잭션 분석 모델은, 기존의 구문 패턴 인식 기법을 활용하여, 위의 페이로드 트랜잭션 확장을 표현하는 트랜잭션의 생성 규칙을 CFG로 정의하고, CFG를 정의하기 위한 페이로드에 트랜잭션 타입을 정의했으며, 해당 생성 규칙을 활용하는 페이로드 트랜잭션 패턴 탐지 기법을 제시하여, 실제 OpenADR 2.0b가 서비스하는 페이로드 트랜잭션 취약점을 탐지하는 사례를 보였다.

본 논문에서 제시한 페이로드 트랜잭션에 개념이 OpenADR 2.0b 프로파일 사양[10]의 이해와 오픈소스 구현에 도움이 될 것이라 기대하고, 어플리케이션 계층에서 페이로드 트랜잭션 분석모델을 사용해 정적분석과 취약점 보안 개발에 도움이 될 수 있으리라 기대한다.

ACKNOWLEDGMENTS

본 논문은 2015년 산업통상자원부의 재원으로 한국에너지기술평가원의 지원을 받아 수행된 것임.(No. 20151220100090)

REFERENCES

[1] JKeun-Ho Lee, “Analysis of Threats Factor in IT Convergence Security”, Journal of the Korea Convergence Society, Vol. 1, No. 1, pp. 49-55, 2013.

[2] Seong-Hoon Lee, Dong-Woo Lee, “Actual Cases for Smart Fusion Industry based on Internet of Thing”, Journal of the Korea Convergence Society, Vol. 7, No. 2, pp. 1-6, 2016.

[3] SBo-Seon Kang, Keun-Ho Lee, “A Scheme on Energy Efficiency Through the Convergence of Micro-grid and Small Hydro Energy”, Journal of the Korea Convergence Society, Vol. 6, No. 1, pp. 29-34, 2015.

[4] United States Department of Energy: Benefits of Demand Response in Electricity Markets and Recommendations for Achieving Them, Technical Report, 2006.

[5] W. M. Taqqali and N. Abdulaziz, “Smart Grid and Demand Response Technology”, 2010 IEEE International, 2010.

[6] P. Faria, Z. Vale, and J. Baptista, “Demand response programs design and use considering intensive penetration of distributed generation”, Energies, Vol. 8, No. 6, 2015.

[7] H. A. Aalami, M. Parsa Moghaddam, and G. R. Yousefi, “Demand response modeling considering interruptible/curtailable loads and capacity market programs”, Applied Energy, Vol. 87, No. 1, pp. 243-250, 2010.

[8] S An, IC Lim, SH Kim, ES Yuk, "Development of a Peak Power Control System based on Zigbee Wireless Communication", Journal of Institute of Control, Robotics and Systems, Vol. 21, No. 5, pp. 442-446, 2015.

[9] OpenADR Alliance: OpenADR 2.0 Profile Specification A Profile, Technical Report, 2011.

[10] OpenADR Alliance: OpenADR 2.0 Profile Specification B Profile, Technical Report, 2013.

[11] Ulrich Herberg, Daisuke Mashima, Jorjeta G. Jetcheva, Sanam Mirzazad-Barijough, “OpenADR 2.0 deployment architectures: Options and implications”, 2014 IEEE International, 2014.

[12] SATO Tetsuji, “The Current Status OpenADR (Automated Demand Response) Technology and NEC’s Approach to the DR Market”, NEC Technical Journal, Vol. 10, No. 2, pp. 87-90, 2016.

[13] OpenADR Alliance: OpenADR 2.0 Demand Response Program Implementation Guide, Technical Report, 2016.

[14] Andrew Paverd, Andrew Martin, Ian Brown, “Security and Privacy in Smart Grid Demand Response System”, International Workshop on Smart Grid Security, 2014.

[15] Hyeon-Ho Chae, June-Kyoung Lee, Kyoung-Hak Lee, “A Study on The Security Vulnerability Analysis of Open An Automatic Demand Response System”, Journal of Digital Convergence, Vol. 14, No. 5, pp. 333-339, 2016.

[16] Mijeong Park, Miyoung Kang, Jin-Young Choi, “The research on vulnerability analysis in OpenADR for Smart Grid”, International Workshop on Data Analytics for Renewable Energy Integration, 2014.

[17] Certified EPRI Open Source OpenADR 2.0b VEN & VTN, http://www.openadr.org/

[18] http://www.openadr.org/faq

[19] Sevket Gokay, Markus C. Beutel, Houran Ketabdar, Karl-Heinz Krempels, “Connecting Smart Grid Protocol Standards: A Mapping Model Between Commonly-used Demand-response Protocols OpenADR and MIRABEL”, 2015 International Conference on. IEEE, 2015.

[20] Tou, Julius T., Rafael C. Gonzalez, “Pattern recognition principles”, Addison-Wesley, 1974.

[21] Fu, King-Sun, Bharat K. Bhargava. “Tree Systems for syntactic pattern recognition”, IEEE Transactions on Computers, Vol. C-22, 1087-1099p, 1973.

저자소개

최 민 영(Choi, Min-Young) [정회원]

- 2014년 2월 : 한양대학교 컴퓨터공학과(공학사)
- 2016년 2월 : 한양대학교 컴퓨터공학과(공학석사)
- 2016년 2월~현재 : ㈜나온웍스 주임 연구원

<관심분야> : 프로그래밍 언어, 정적 분석, 네트워크, 융합 보안

이 준 경(Lee, June Kyoung) [정회원]

- 1993년 2월 : 인하대학교 전자계산공학과(공학사)
- 1995년 2월 : 인하대학교 전자계산학과(공학석사)
- 2000년 8월 : ㈜LG 정보통신 선임 연구원
- 2007년 7월~현재 : ㈜나온웍스 대표

<관심분야> : 네트워크, 프로토콜, 융합 보안

이 경 학(Lee, Kyoung Hak) [정회원]

- 1992년 2월 : 광운대학교 전자통신공학과(공학사)
- 1994년 2월 : 광운대학교 전자통신공학과(공학석사)
- 2007년 2월 : 광운대학교 전자통신공학과(공학박사)
- 1994년 4월~2011년 12월 : 한국산업기술평가관리원 책임
- 2012년 3월~2016년 8월 : 남서울대학교 조교수
- 2016년 9월~현재: 광운대학교 부교수

<관심분야> : VR, S/W platform, 융합 보안