Journal of Korea Multimedia Society Vol. 25, No. 6, June 2022(pp. 793-806)
https://doi.org/10.9717/kmms.2022.25.6.793

ISO 19847 선박 데이터 서버 이중화를 통한 사이버 보안 취약성 대응 방안 실험

이창의†, 이서정††

Experiment on countermeasures against cyber security vulnerabilities using redundancy of ISO 19847 Shipboard Data Server

ChangUi Lee†, Seojeong Lee††

ABSTRACT

As the IMO introduced MASS (Maritime Autonomous Surface Ships), ISO(International Organization for Standardization) announced ISO 19847 of a maritime data sharing standard for collecting and remotely managing data of ship systems. Previous literature evaluated the risk using HAZOP for ISO 19847 and proved that risk assessment is useful through experiments. However, redundancy of ISO 19847 ship data server which is one of the risk reduction method suggested in previous literature, was designed but couldn't tested due to the limitations of the conditions. So, in this study, to prove the usefulness of the ship data server redundancy of ISO 19847 which was not tested in previous literature. It based on the design of previous literature, and the network of ship data servers was modeled using the SES/DEVS format and simulated using the DEVS# open source library.

Key words: ISO 19847, Shipboard Data Server Redundancy, Cyber Security, SES(System Entity Structure), DEVS(Discrete Event System Specification)

1. 서 론

ICT 융합기술이 선박과 해양분야에 빠르게 적용되면서 유럽 국가들과 중국 및 일본을 비롯한 많은 나라가 차례로 자율운항선박 계획을 발표하고 있으며, 각 나라와 회사들이 미래의 핵심 사업으로 중점적으로 개발하고 있다[1,2,3]. 이에 따라 국제표준화기구(ISO, International Organization for Standardization)에서는 선박 시스템들의 데이터를 수집하고 원격지에서 관리하기 위한 해상데이터 공유 표준인 ISO 19847 선박 및 해양기술-해상에서 필드 데이터 공유를 위한 선박 데이터 서버(Ships and marine technology -Shipboard data servers to share field data at sea)를 2018년 10월에 제정하였다[3,4,5].

C.U. Lee et al.은 ISO 19847/19848이 산업현장에 적용되기에 앞서 기능 안전성 측면에서 위험성 분석을 수행하여 잠재적인 문제점을 도출하기 위해 ISO 19847의 선박 데이터 서버(Shipboard data server)를

※ Corresponding Author : Seojeong Lee, Address: (49112) No. 318, College of Maritime Sciences (No. C6), Korea Maritime and Ocean University 727, Taejong-ro, Yeongdo-Gu, Busan, South Korea, TEL : +82-51-410-4578, FAX : +82-51-410-3985, E-mail : sjlee@kmou.ac.kr
Receipt date : May 17, 2022, Revision date : Jun. 2, 2022
Approval date : Jun. 10, 2022

† Division of Computer Engineering, Graduate School, Korea Maritime & Ocean University
(E-mail : myallyou@gmail.com)
†† Division of Maritime System Engineering, Korea Maritime & Ocean University
※ This research is a part of "AI-based heavy cargo ship logistics platform demonstration project " hosted by the Ulsan ICT Promotion Agency, supported by the National IT Industry Promotion Agency and the Ministry of Science and ICT.' (Project number: S1510-22-1001)

대상으로 소프트웨어 기능 안전성 측면에서 HAZOP (Hazard and Operability Study) 방식의 위험성 분석을 수행하여 25종의 일탈을 도출하고 위험등급을 부여하였다[6]. 그리고 도출된 위험 중에서 위험등급이 높은 11가지의 일탈과 원인 및 안전대책을 정리하여 분석해 IEC 61508-3의 반정형기법을 통해 ISO 19847의 선박 데이터 서버의 구현에 있어 4종의 안전대책을 제시하였다[7]. 그 뒤 제시한 4가지 안전대책에 따라 소프트웨어 아키텍처를 수정하고 실험하여 안전대책이 실제로 효과가 있는지를 확인하였다[8]. 하지만, 4가지 안전대책 중에서 선박 데이터 서버의 이중화는 설계는 하였으나 여건의 한계로 실험하지 못하였다. 이 연구에서는 선행 연구의 서버 이중화 설계를 기초로 사이버 보안 취약성 공격 시나리오를 이용하여 선박 데이터 서버의 이중화 설계에 대해 시뮬레이션을 통해 검증하고자 한다. 이를 위하여 시뮬레이션 기반 환경으로 이산사건 시스템 모델링 방법으로 많이 활용되고 있는 SES(System Entity Structure) / DEVS(Discrete Event System Specification) 모델링을 통하여 접근하였다.

2. ISO 19847 선박 데이터 서버

ISO 19847은 해상 필드 데이터 공유를 위한 선박 데이터 서버에 관한 국제표준으로 선박 내 시스템에서 생성된 데이터를 저장하고 육상으로 전달하는 데이터 서버로서 기능, 성능, 서비스 및 안전요건에 대해 제시하고 있다. 또한, ISO 19848은 선박시스템과 데이터 서버간 데이터 교환에 관한 표준으로 선박에서 데이터를 교환하고 처리하는 것을 용이하도록 하기 위해 시스템을 분류하여 구분할 수 있는 데이터 규칙 및 식별자를 정의하고 있다[4,6].

Fig. 1은 두 표준의 개념모델로써 (A) 부분은 ISO 19847의 개념모델로 입력기능, 출력기능, 데이터 저장소로 구성된다. 수신된 데이터는 기본적으로 저장소에 저장되도록 정의하고 있으며 스트리밍으로 받은 데이터를 중계하여 스트리밍 서비스를 수행하고 요청에 따라 저장된 데이터를 검색하여 응답하는 서비스를 수행한다. (B) 부분은 ISO 19848의 개념모델로 입력데이터와 출력데이터의 형식에 대해서 정의하고 있다. 계층적 구조를 통해 데이터를 추상화하고 데이터의 형식과 종류, 범위 등을 정의할 수 있도록 데이터 구조를 정의하고 있다[6,9].

선박 데이터 서버는 IEC 61162-1/2/450 등의 데이터를 입력받아 데이터를 전달하고, 저장하는 기능을 수행한다. 데이터 전달을 위한 출력기능으로 스트리밍 방식과, 요청-응답 방식, 파일 전송방식을 가진다. 스트리밍 방식을 통한 프로토콜은 MQTT를 권고하고 있으며, 요청-응답 방식은 HTTP 프로토콜을, 파일 전송 방식은 FTP 프로토콜을 사용하는 것을 권고하고 있다.

C.U. Lee et al.은 ISO 19847의 선박 데이터 서버의 위험성 평가를 통해 Fig. 2에서와 같이 데이터 입력, 데이터 출력, 전송 주기, 전송 속도, 통신 신뢰성, 통신 안정성 측면에서 위험 등급이 높은 위험들을 식별하였고, 4종의 안전대책을 제시하였다. 제시한 안전대책은 선박의 데이터가 많이 발생함으로써 생기는

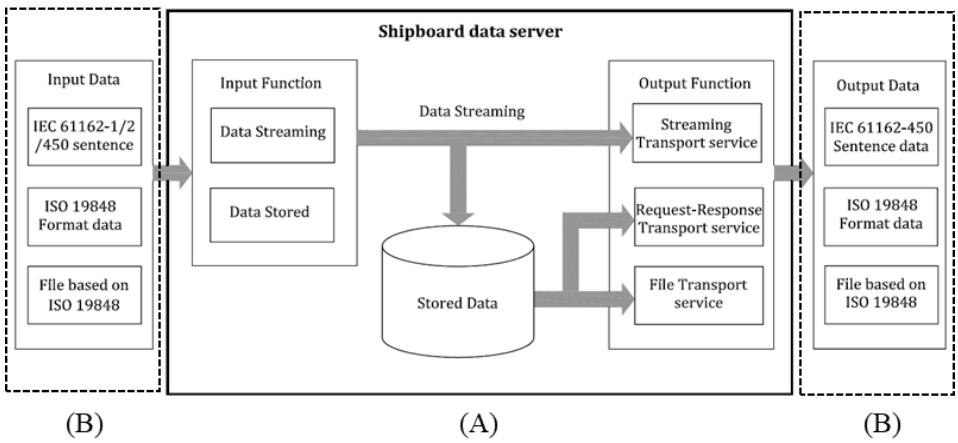

Fig. 1. ISO 19847/19848 concept model.

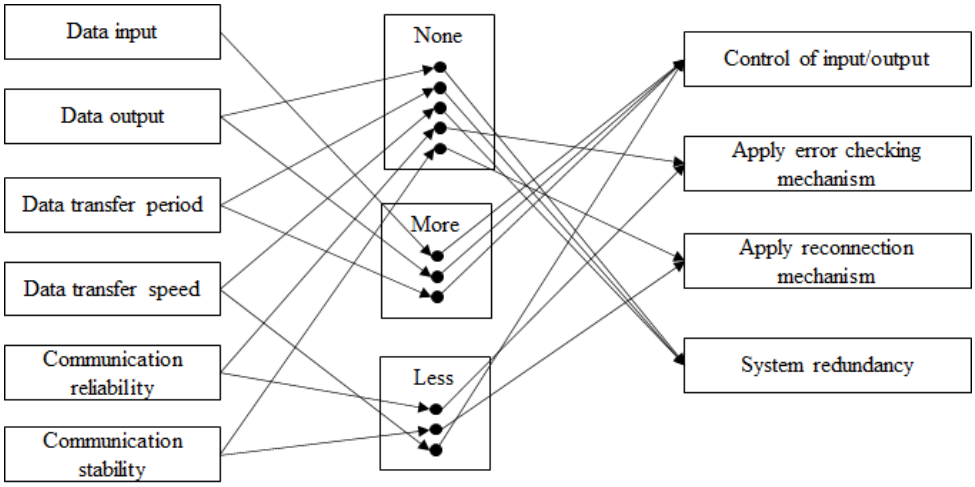

Fig. 2. High risk deviations, causes and safeguards in previous literature.

문제를 방지하기 위한 데이터 입/출력량 조절, 선박과 육상의 신뢰성 있는 통신을 위한 오류 확인 메커니즘 적용, 선박의 음영지역 진입에 의한 문제를 방지하기 위한 재접속 메커니즘 적용, 단일의 선박 서버가 문제가 발생할 경우를 대비한 시스템 이중화가 있다[7]. 안전대책에 따라 데이터 입/출력량 조절과 오류 확인 메커니즘 적용, 재접속 메커니즘 적용을 위하여 소프트웨어 아키텍처를 수정하고 인말새트 FB-250 위성통신 시스템을 사용하여 실험하여 안전대책이 실제로 효과가 있는지를 확인하였다. 하지만, 선박 데이터 서버의 이중화는 Fig. 3과 같이 네트워크 토폴로지를 구성하고, Fig. 4와 같이 선박 데이터 서버의 상태를 지속적으로 확인하여 실패가 감지되면 대기 중이던 다른 서버에 트래픽을 전달하도록 설계를 하였으나 여건의 한계로 실험하지 못하였다[8]. 그래서 이 연구에서는 선행 연구에서 위험성 분석을 통해 식별된 단일 선박 데이터 서버의 문제가 발생할 경우를 대비한 위험성과 이를 해결하기 위한 선박 데이터 서버의 이중화 설계를 기반으로 하여 시뮬레이션을 통해 선행 연구의 안전대책이 효과가 있음을 확인하고자 한다.

3. SES 및 DEVS 형식론

SES 형식론은 트리 구조를 이용하여 하나의 시스템이 가지는 대안들을 총체적으로 표현하는 형식론으로 Zeigler가 제안한 개념이다[10,11]. SES에는 시스템의 구조를 표현하기 위한 지식으로 Entity, Aspect, Specialization의 3가지 형태의 노드가 있다. Entity 노드는 실세계의 한 객체와 대응되며 Aspect 또는 Specialization을 자식으로 가질 수 있다[[10]. 일반적으로 시스템은 SES의 서브-구조(Sub-Structure)로

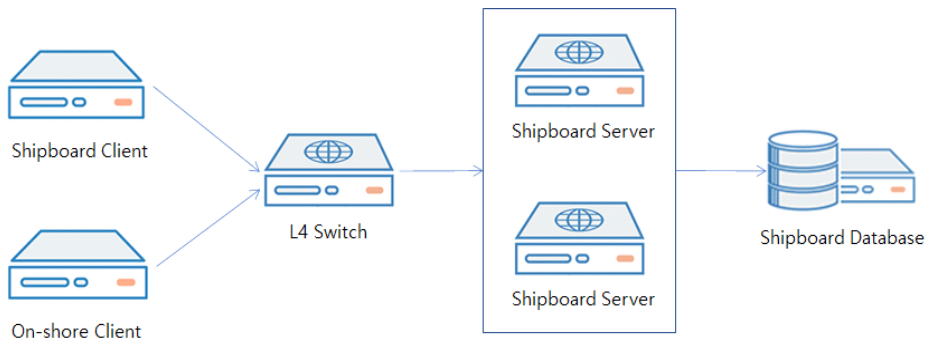

Fig. 3. Architecture of system redundancy in previous literature.

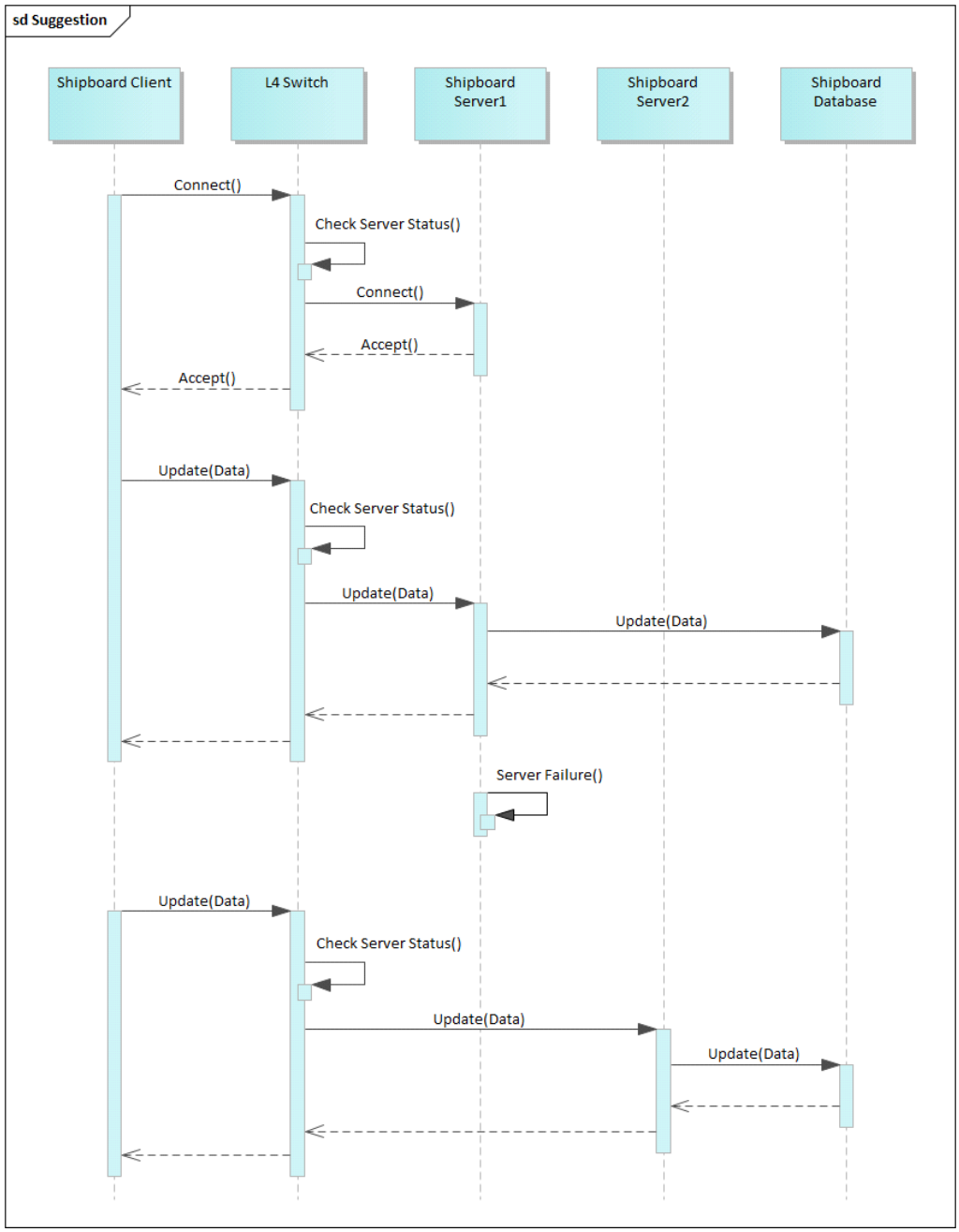

Fig. 4. Error handling procedure by system redundancy in previous literature.

표현된다. 이 서브-구조를 PES(Pruned Entity Structure)라고 하며 SES에서 PES를 얻어내는 작업을 전정(Pruning)이라고 한다[11,12]. PES에는 오직 Entity와 Aspect만이 존재한다. 즉, Specialization 노드의 자식 노드는 전정 과정에서 하나의 노드만 선택되고 나머지 Specialization 노드는 사라진다[11,13,14].

SES는 트리(Tree) 형태로 표현되며 관계를 나타내는 간선은 그 선의 종류(단선, 2중선, 3중선)로 연관관계의 특성을 구분하는데 그 내용은 Table 1과 같다.

Table 1. Component of system entity structure.

Notation	Name	Description
|	Aspect	Decomposition relationship of an entity
||	Specialization	Aaxonomy relationship of an entity
|||	Multiple Aspect	Multiplicity relationship of an entity

SES 형식론을 이용하면 여러 가지 대안에 대하여 전체적으로 모델링을 한 후에 각각을 시뮬레이션함으로써 문제 해결 기법에 유용하다. 또한 방법론적인 시스템을 모델링이 가능하고, 모델의 구조변경이 쉬우며, 기존에 개발된 모델들을 재사용하기에 편리하다. 본 논문에서는 네트워크 보안 시뮬레이션에 SES 형식론을 적용함으로 이와 같은 효과를 기대할 수 있다[10,12].

DEVS 형식론은 1976년 B.P.Zeigler에 의해 제안된 집합론에 근거한 형식론이다. DEVS 형식론은 복잡한 시스템을 구성요소 별로 나누어 각각의 모델을 만든 후, 이를 합쳐서 전체 시스템을 표현할 수 있도록 한다. DEVS에서는 두 가지 모델 유형인 원자 모델(Atomic Model)과 결합 모델(Coupled Model)로 제공하는데, 원자모델은 시스템의 동적인 특성을 표현하기 위한 것이고, 결합모델은 시스템의 구성 요소간의 상호 작용을 표현하기 위한 것이다. 여기서 결합모델은 더 큰 단위의 상위 모델의 기준에서는 구성요소의 단위로 사용이 가능하여 시스템을 계층적으로 표현할 수 있게 한다[11,15,16,17]. 원자 모델의 명세는 Table 2와 같이 정의된다. 여기서, e는 마지막 상태전이 이후로 진행된 시간을 나타낸다. 결합모델은 하위 모델들을 이벤트, 외부입력, 외부출력 등을 연결하여 계층적인 관계를 구성하고, 표현법은 Fig. 5와 같이 정의된다[11].

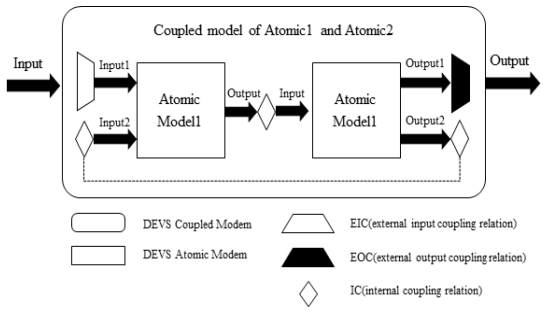

Fig. 5. DEVS Coupled Model Representation.

4. 선박 데이터 서버 이중화 모델링

현실 세계에서 네트워크 시스템은 다양한 네트워크 구성요소와 그 구성요소로 이루어진 네트워크들로 구성되어 있다. 본 연구에서는 공격자가 선박 데이터 서버를 어떻게 공격하는지를 확인하여 C.U. Lee et al.의 선행 연구에서 실험하지 못한 선박 데이터 서버의 이중화에 대한 검증이 목적이므로 네트워크 시스템을 단순화시켜 Fig. 6과 같이 네트워크 토폴로지를 구성하였다.

Fig. 6에서 보는 바와 같이 네트워크는 1개의 방화벽과 1개의 L4 스위치, 1개의 호스트 기반 침입 탐지

Table 2. Definition of atomic model.

Notation	Name	Description
M	Component model set	M=<X, Y, S, Y, δ_{int}, δ_{ext}, λ, ta>
X	Input event set	
Y	Output event set	
S	Sequential state set	
δ_{int}	Internal transition function	S → S
δ_{ext}	External transition function	Q × X → S
Q	Set of all states	$Q=(s,e \mid s \in S,\ 0 \le e < ta(s)$
λ	Output function	S → Y
ta	Time advance function	S → $R^{+}_{0,\infty}$

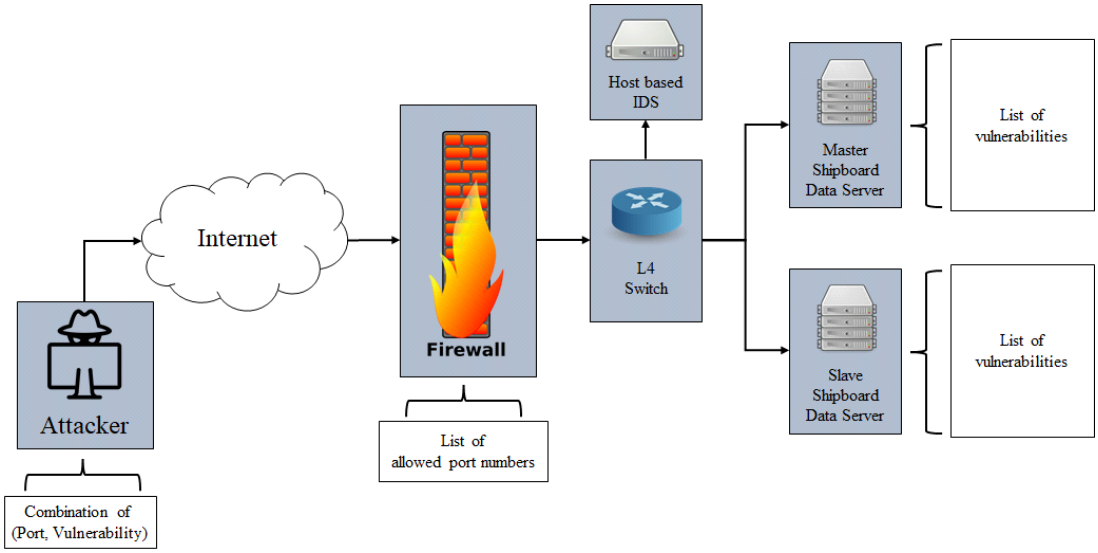

Fig. 6. Network Topology.

시스템(Intrusion Detection System, IDS), 2개의 선박 데이터 서버로 구성되어 있으며, 방화벽은 허용되는 포트 목록을 제외한 모든 포트에서 패킷을 차단하도록 하고 선박 데이터 서버는 사용자에게 서비스를 제공하고 몇몇 취약성을 내포하고 있다. 공격자는 포트와 서비스를 무작위로 조합하여 서버 시스템의 취약성을 공격할 것이며, 침입 탐지 시스템은 학습된 공격자의 공격 패턴을 통해 침입을 감지할 것이다. Fig. 7은 본 연구를 위하여 단순화된 네트워크 토폴로지를 SES를 활용하여 모델링한 결과이다.

네트워크는 클라이언트와 방화벽, 스위치, 서버로 구성되어 있으며 각 구성 요소들은 복수개로 구성될 수 있다. 클라이언트는 일반적으로 서비스를 사용하는 노드와 공격자로 구분할 수 있고, 방화벽은 패킷 필터, 서킷 게이트웨이, 프록시 서버로 구성될 수 있다. 또한 스위치는 L3 스위치, L4 스위치로 구성될 수 있으며, 침입 탐지 시스템은 네트워크 기반과 호스트 기반 침입 탐지 시스템으로 구성될 수 있다. 서버는 선박 데이터 서버, 웹서버, 메일서버, DB서버로 구성될 수 있다.

시뮬레이션을 위하여 Fig. 7의 SES를 Specialization 노드의 자식 노드 중 하나의 노드만 선택하고 나머지 노드는 삭제하는 전정(가지치기)를 통해 Entity와 Aspect만이 존재할 수 있는 PES를 Fig. 8과 같이 클라이언트 중에서는 공격자를, 방화벽 중에서는 패킷 필터를, 스위치는 L4 스위치를, 침입 탐지 시스템에는 호스트 기반 침입 탐지 시스템을, 서버는 선박 데이터 서버를 선택하여 모델링 하였다.

Fig. 7과 Fig. 8의 SES와 PES를 통해 시스템이 구조적인 부분을 모델링하였다. 시스템의 동적인 모

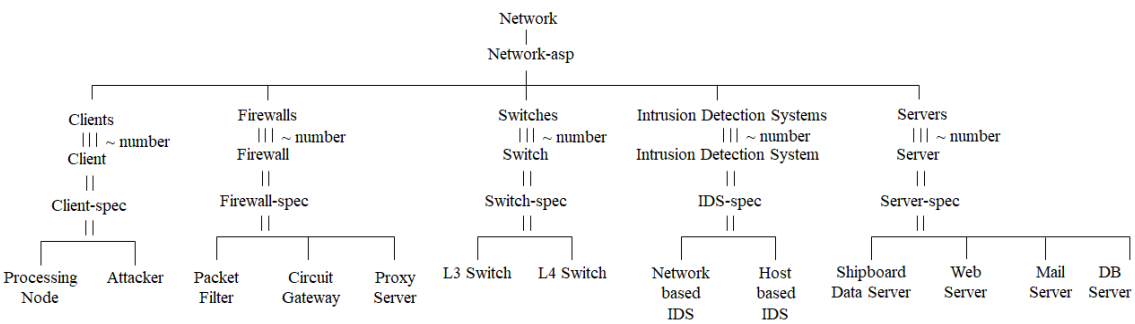

Fig. 7. Network SES.

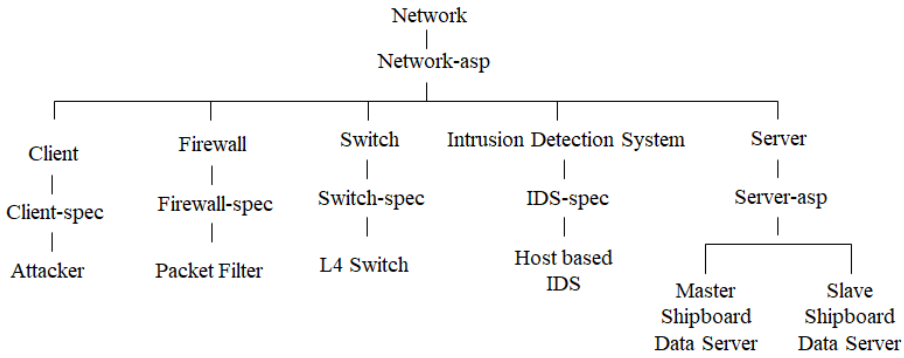

Fig. 8. Network PES.

델링을 위하여 Fig. 9에서 보는 바와 같이 시스템의 구조적 모델링을 통해 정의되었던 공격자와 방화벽, L4 스위치, 호스트 기반 침입 감지 시스템, 마스터 및 슬레이브 선박 데이터 서버의 입출력 포트를 정의하고, 공격 메시지를 생성하는 발생기를 추가하여 시뮬레이션 구성도를 작성하였다.

Fig. 9에서 공격자는 악의적인 목적을 가지고 취약성을 이용하여 공격하는 사람을 말하고, 방화벽은 외부의 데이터를 내부로 통과시킬지를 결정하는 장치이다. L4 스위치는 내부 네트워크의 상태에 따라 데이터를 어떤 경로로 누구에게 전달할 것인지를 결정하는 장치이다. 마스터 선박 데이터 서버와 슬레이브 선박 데이터 서버는 선박의 데이터를 취합하여 육상으로 전달하는 시스템이다. 그리고 발생기는 시뮬레이션 전체를 관리하며 공격자에게 공격을 시작하도록 하는 이벤트를 발생시키는 역할을 한다.

모델링된 7개의 컴포넌트(발생기, 공격자, 방화벽, L4 스위치, 호스트 기반 침입 감지 시스템, 마스터 선박 데이터 서버, 슬레이브 선박 데이터 서버)와 시뮬레이션을 제어하는 제어기의 입력과 출력에 따른 동작을 알아보기 위하여 Fig. 10에서와 같이 상태 전이도를 구성하였다.

제어기는 활성 상태에서 시뮬레이션을 제어하다가 결과가 수신되면 결과를 출력하고 대기상태로 전환된다. 발생기는 바쁨상태로 있다가 출력 메시지를 전송하고 다시 바쁨 상태에 머무르다 지정된 공격횟수에 다다르면 대기상태로 전환된다. 공격자는 대기상태에 있다가 입력이 들어오면 바쁨상태로 전환되어 공격을 수행한다. 수행한 결과가 수신되면 수신됨 상태로 있다가 결과를 출력하고 대기상태로 돌아간다. 방화벽과 L4 스위치, 호스트 기반 침입 감지 시스템은 대기상태에 있다가 입력이 들어오면 바쁨상태에서 필요한 동작을 수행하고 적절한 출력을 내보낸 다음 다시 대기상태로 돌아간다.

마스터 선박 데이터 서버와 슬레이브 선박 데이터 서버는 C.U. Lee et al.의 선행 연구의 설계에 따라 Active-Standby 형태로 동작한다. 그에 따라 마스터 선박 데이터 서버는 대기상태에 있다가 입력이 들어오면 바쁨상태에서 필요한 동작을 수행하고 적절한 출력을 내보낸 다음 다시 대기상태로 돌아간다. 비활

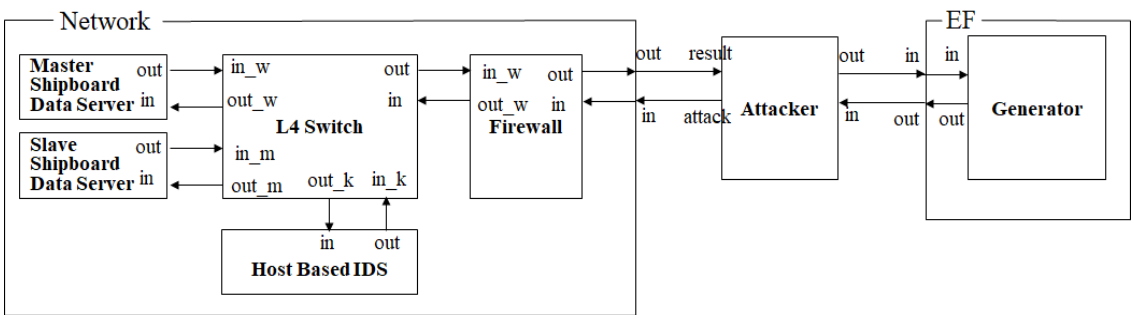

Fig. 9. Simulation Diagram.

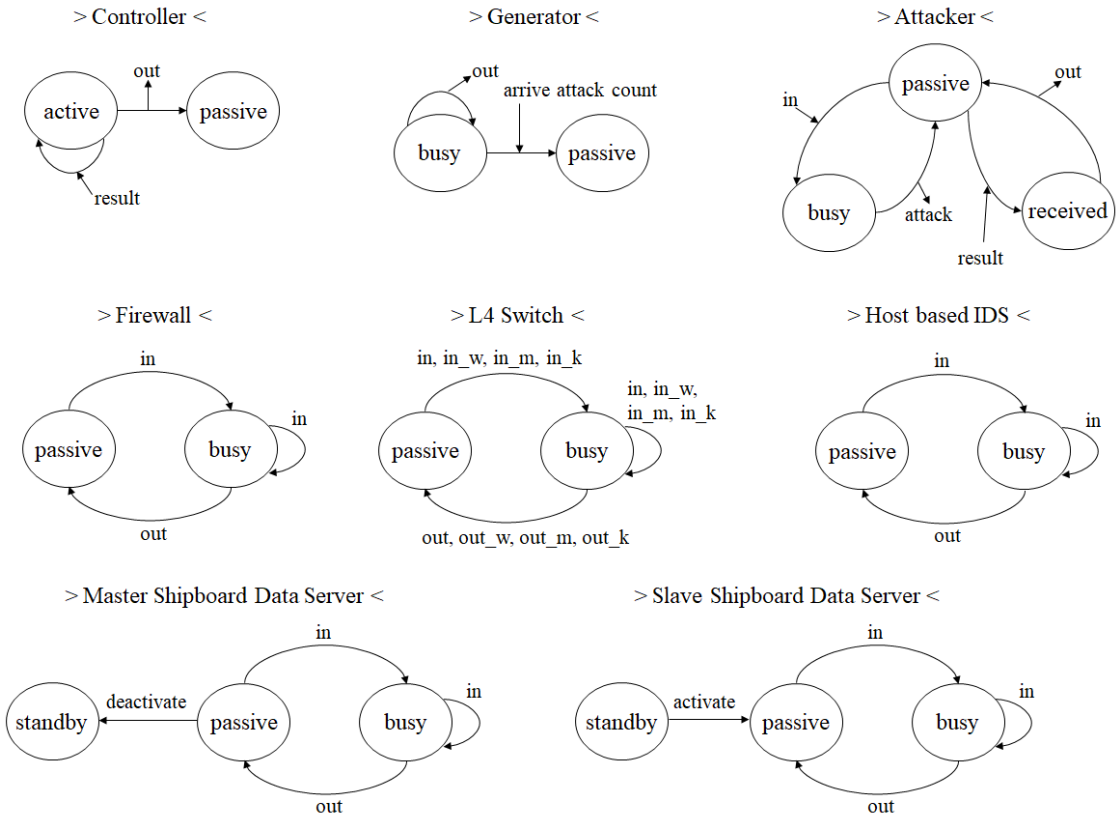

Fig. 10. Simulation component state-transition diagram.

성화 명령이 수신되면 준비상태로 전이된다. 그리고, 슬레이브 선박 데이터 서버는 준비상태에 있다가 활성화 명령이 수신되면 준비 동작을 수행하고 대기상태에서 머문다. 그 후 입력이 들어오면 바쁨상태에서 필요한 동작을 수행하고 적절한 출력을 내보낸 다음 다시 대기상태로 돌아간다. 시뮬레이션 구성도와 상태 전이도를 바탕으로 각 컴포넌트의 DEVS 형식론으로 표현하면 Table 3과 같다.

5. DEVS#을 이용한 시뮬레이션

DEVS#은 DEVS-C++과 더불어 이산사건 기반 시뮬레이션에서 많이 사용되고 있는 C#언어로 DEVS를 구현한 오픈 소스 라이브러리이다. 본 연구에서는 가비지 수집, 유형 검사 기능, 웹 기능 등과 같은 C++보다 몇 가지 장점이 있는 DEVS#을 이용하여 시뮬레이션을 수행하였다[13,15].

5장에서 구조적/동적 모델링 한 것을 정리하여 보면, (1) 제어기는 시뮬레이션을 제어하고, (2) 발생기는 공격자에게 공격 시작 메시지를 전달하여 (3) 공격자는 공격하고자 하는 포트와 서비스를 조합하여 네트워크에 공격을 가한다. (4) 방화벽은 허용 포트 목록에서 대상 포트를 조회하여 내부로 패킷을 전달할 것인지를 판단하여 패킷을 전달하고, (5) L4 스위치는 선박 데이터 서버 중 어느 서버로 보낼지 결정한다. (6) 동시에 호스트 기반 침입 감지 시스템은 수신된 패킷을 검사하여 공격 여부를 판단한다. (7) 선박 데이터 서버는 요청된 서비스가 서비스 목록과 취약성 목록 중에서 어느 곳에 해당하는지를 확인하여 결과를 응답한다. 이때 공격자가 서버로부터 취약성 결과를 받게 된다면 공격이 성공한 것을 의미한다.

취약성에 대한 목록은 미국 연방 정부의 후원을 받아 비영리 연구 개발 기관인 MITRE가 소프트웨어와 펌웨어의 취약점들을 파악하고 분류해 모아놓은 CVE(Common Vulnerabilities and Exposures) 웹사이트를 참조하여 ISO 19847 선박 데이터 서버가 지원하는 3종의 서비스(MQTT, HTTP, FTP)에 대

Table 3. DEVS Formalism of simulation component.

Component	DEVS Formalism	Component	DEVS Formalism
Controller	X = { stop } Y = { out } S = { active, passive } δint(active) = active δext(busy, e, stop) = passive λ(active) = out ta(passive) = ∞ ta(busy) = 0	L4 Switch	X = { in, in_w, in_m, in_k } Y = { out, out_w, out_m, out_k } S = { passive, busy } δint(busy) = busy δext(passive, e, in) = busy λ(busy) = out ta(passive) = ∞ ta(busy) = 0
Generator	X = { stop } Y = { out } S = { passive, busy } δint(busy) = busy δext(busy, e, stop) = passive λ(busy) = out ta(passive) = ∞ ta(busy) = 0	Host based IDS	X = { in } Y = { out } S = { passive, busy } δint(busy) = busy δext(passive, e, in) = busy λ(busy) = out ta(passive) = ∞ ta(busy) = 0
Attacker	X = { in, result } Y = { out, attack } S = { passive, busy, received } δint(busy) = busy δint(received) = passive δext(passive, e, in) = busy δext(busy, e, result) = received λ(busy) = out ta(passive) = ∞ ta(busy) = 0 ta(result) = 0	Master shipboard data server	X = { in } Y = { out } S = { passive, busy, standby } δint(busy) = busy δext(passive, e, in) = busy δext(passive, e, deactivate) = standby λ(busy) = out ta(passive) = ∞ ta(busy) = 0
Firewall	X = { in } Y = { out } S = { passive, busy } δint(busy) = busy δext(passive, e, in) = busy λ(busy) = out ta(passive) = ∞ ta(busy) = 0	Slave shipboard data server	X = { in } Y = { out } S = { standby, passive, busy } δint(busy) = busy δext(standby, e, activate) = passive δext(passive, e, in) = busy λ(busy) = out ta(passive) = ∞ ta(busy) = 0

한 취약성을 추출하여 작성하였으며, 추출된 취약성 목록은 Table 4와 같다[14,18].

Table 4에서 CVE-2022-25137와 같은 취약성 번호는 보안 취약점을 발견하여 취약성 목록 관리기관인 MITRE에 보고하면, CVE라는 문자에 취약점이 발견된 연도와 임의의 번호를 붙여 만들어진다. CVE-

Table 4. List of Extracted Vulnerabilities.

Service Type	Vulnerabilities
MQTT	CVE-2022-25137, CVE-2022-25136, CVE-2022-25135, CVE-2022-25134, CVE-2022-25133, CVE-2022-25132, CVE-2022-25131, CVE-2022-25130, CVE-2021-41039, CVE-2021-41036
HTTP	CVE-2022-29942, CVE-2022-29491, CVE-2022-29180, CVE-2022-29167, CVE-2022-28994, CVE-2022-28711, CVE-2022-28708, CVE-2022-28561, CVE-2022-28560, CVE-2022-28380
FTP	CVE-2022-29051, CVE-2022-29050, CVE-2022-28157, CVE-2022-26130, CVE-2022-23135, CVE-2022-22989, CVE-2022-22899, CVE-2021-43774, CVE-2021-42110, CVE-2021-40524

2022-25137은 2022년에 보고된 명령 주입 취약점으로, 공격자가 조작된 MQTT 패킷을 통해 임의의 명령을 실행할 수 있는 취약점에 대해 관리 번호를 부여한 것이다.

시뮬레이션을 수행하기 위한 시뮬레이터를 개발하기 위해 Fig. 11과 같이 클래스를 다이어그램을 작성하였다. MainFrame 클래스는 시뮬레이터 프로그램의 화면 구성과 사용자의 버튼 클릭에 대한 동작을 정의하고 있으며, Controller 클래스를 포함한 8개의 클래스는 Table 3에서 DEVS 형식론에서 정의한 동작을 코드로 구현하였다.

시뮬레이션을 위하여 MS Visual Studio 2022와 DEVS# 라이브러리를 이용하여 Fig. 6의 네트워크 토폴로지를 기반으로 Fig. 12와 같이 화면을 구성하였다.

시뮬레이터 화면은 공격자의 공격 시나리오 리스트와 방화벽의 허용된 포트 목록, L4 스위치, 호스트 기반 침입 감지 시스템, 선박 데이터 서버의 취약성

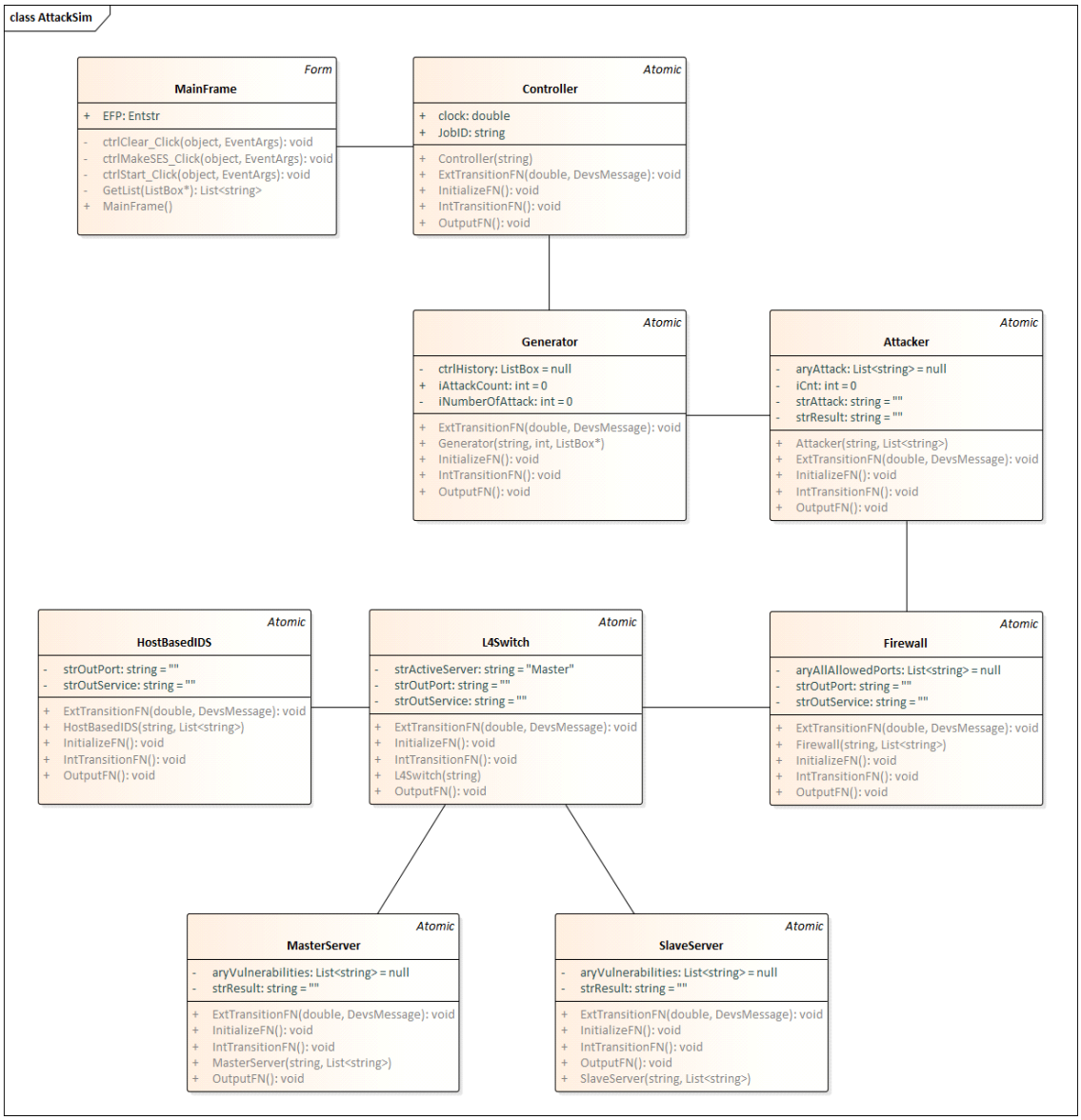

Fig. 11. Class diagram of simulator.

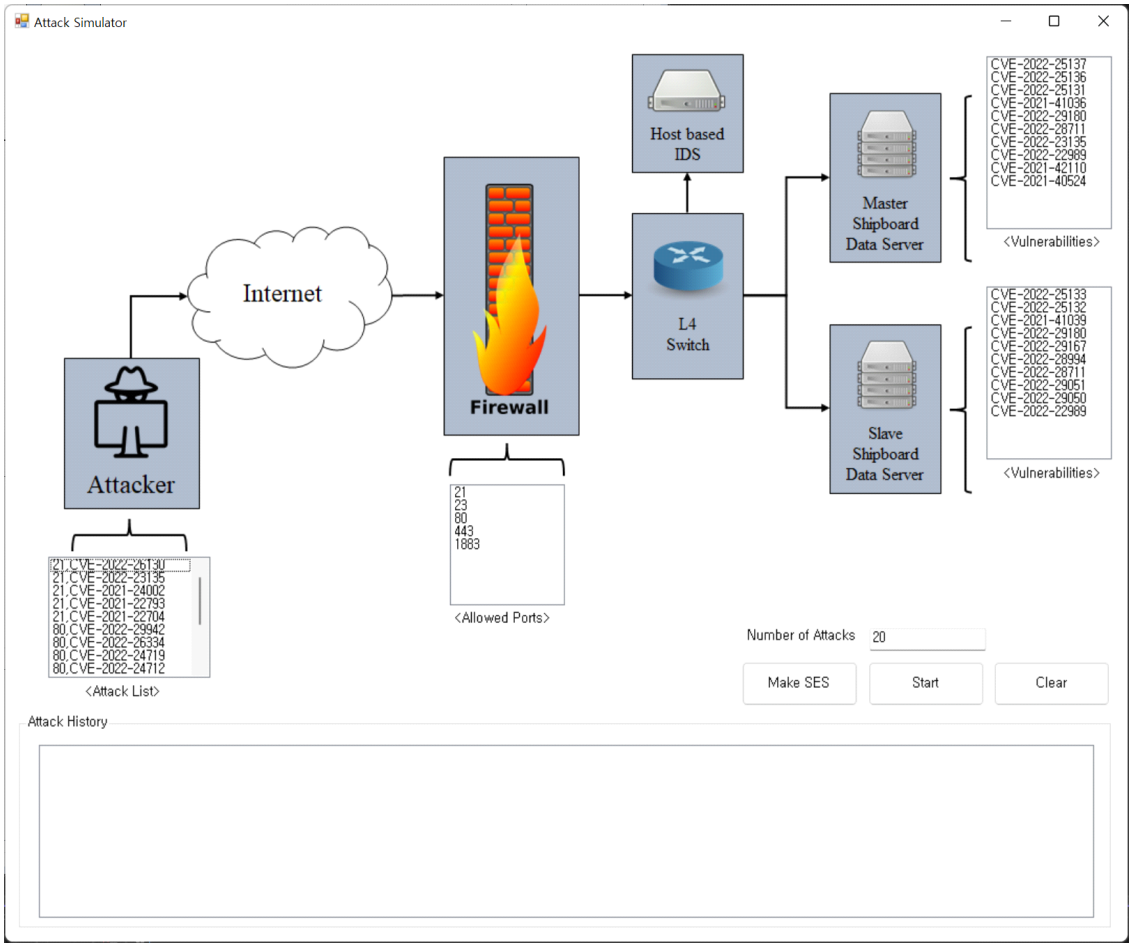

Fig. 12. screenshot of simulation configuration.

목록으로 구성되어 있다. 또한, 시뮬레이션을 실행하기 위한 버튼과 공격 횟수를 지정하기 위한 부분, 공격 시뮬레이션 기록을 목록화하여 확인할 수 있는 부분으로 화면을 구성하였다.

시뮬레이션 프로그램 제작을 마치고 시뮬레이션 테스트를 수행하였다. 마스터 선박 데이터 서버와 슬레이브 선박 데이터 서버는 서로 다른 취약성을 가지도록 취약성 목록을 설정하였으며, 방화벽은 MQTT, HTTP, FPT 서비스를 위해 21번, 23번, 80번, 443번, 1883번 포트를 통과하도록 설정하였다. 공격은 총 20회에 걸쳐서 공격 시나리오 리스트 중에서 무작위로 선정하여 수행하게 하여 시뮬레이션할 때마다 공격의 성공 횟수와 그 결과가 변경된다.

Fig. 13은 20회의 공격 중에서 5번째 공격에서 CVE-2022-25137 취약성을 이용하여 1883번 포트를 통해 방화벽을 통과하고, L4 스위치를 통해 마스터 선박 데이터 서버가 가진 취약성을 이용하여 공격에 성공하였다. 성공된 공격으로 인하여 마스터 선박 데이터 서버를 통한 데이터 탈취 및 조작을 할 수 있으므로, 취약성 공격을 호스트 기반 침입 감지 시스템이 인지하고 서비스를 제공하는 서버를 마스터 선박 데이터 서버에서 슬레이브 선박 데이터 서버로 변경하였음을 확인할 수 있다. 그 후 공격자의 공격이 있었음에도 슬레이브 선박 데이터 서버는 공격에 대한 취약점이 존재하지 않아 정상적으로 서비스를 제공하고 있음을 알 수 있다.

6. 결 론

본 연구에서는 네트워크 보안 시뮬레이션을 위하

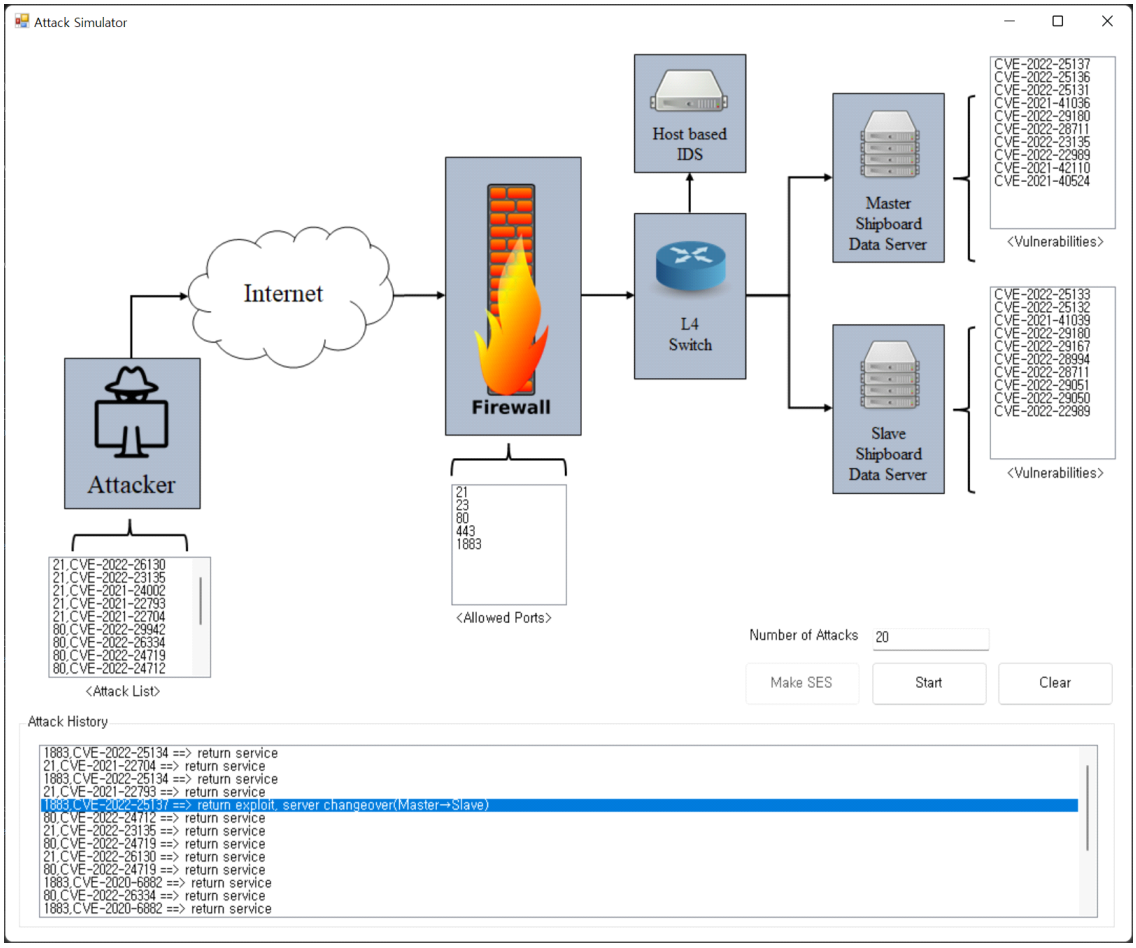

Fig. 13. Result of simulation.

여 구조적 보안 모델링 방법론인 SES/PES와 동적 모델링 방법론인 DEVS를 이용하여 ISO 19847 선박 데이터 서버가 이중화된 네트워크를 모델링하였다. 이중화된 네트워크모델을 오픈 소스 라이브러리인 DEVS#을 이용하여 시뮬레이터를 구성하고, 네트워크 상에 공격자가 취약성을 이용하여 공격하는 형태를 시뮬레이션 테스트하였다. 시뮬레이션 테스트 결과 공격자가 취약성을 통해 공격에 성공하였지만, 서버 이중화를 통해 취약성이 존재하는 서버는 비활성화되고 다른 서버가 대체하여 서비스를 제공함으로써 피해를 방지할 수 있었다.

본 연구에서는 보안 시뮬레이션을 통해 ISO 19847 선박 데이터 서버에 대한 사이버 보안의 필요성과 이중화를 통해 시스템의 견고성(Robustness)을 높일 수 있음을 확인하였다. 다만, 이 연구에서는 네트워크 구성을 단순화하였고, 취약성을 코드화하여 간략히 표현하여 실험하였으나, 향후 현실과 더욱 유사한 네트워크 구성과 공격자가 취약점을 악용하여 최종 목표를 달성하기까지를 구성할 수 있도록 보완해야 할 것이다. 또한, 시뮬레이션에서 선박 데이터 서버는 서로 다른 취약성을 가지고 있어 이중화를 통한 피해 방지가 되었으므로 실제로 선박 데이터 서버를 이중화할 때도 구축환경을 서로 다르게 하여 이중화된 선박 데이터 서버가 비슷한 취약성을 가지지 않도록 해야 할 것이다.

REFERENCE

[1] F. Goerlandt, “Maritime Autonomous Surface Ships from a Risk Governance Perspective: Interpretation and Implications,” *Safety Sci-*

ence, Vol. 128, 2020.

[2] Ø.J. Rødseth and H.N. Wennersberg, "Towards Approval of Autonomous Ship Systems by Their Operational Envelope," *Journal of Marine Science and Technology*, Vol. 27, pp. 67-76, 2022.

[3] G.I. Lee and S.W. Hwang, "Trends in International Standardization of Maritime Autonomous Surface Ships," *Telecommuni- cations Technology Association*, Vol. 175, pp. 115-122, 2018.

[4] ISO, *Ship and Marine Technology - Shipboard Data Servers to Share Field Data at Sea*, ISO Std. 19847:2018, 2018.

[5] ISO, *Ship and Marine Technology - Standard Data for Shipboard Machinery and Equipment*, ISO Std. 19848:2018, 2018.

[6] C.U. Lee and S. Lee, "Design of ISO/IEC 19847 Ship Data Server Based on Functional Safety Analysis," *Proceeding of the Conference of the Korean Society of Marine Engineering*, 2021.

[7] C.U. Lee and S. Lee, "Adjustment Needs on ISO/IEC 19847 for Ship Data Server Implementation," *Proceeding of the Spring Conference of the Korean Society of Marine Environment & Safety*, 2021.

[8] C.U. Lee and S. Lee, "Implementation of ISO/IEC 19847 Ship Data Server Applied Functional Safety," *Proceeding of the Summer Conference of Digital Contents Society*, pp. 93-95, 2021.

[9] Y.T. Woo, et. al, "A Development of Data Management Platform for Shipboard Machinery Equipment to Share Maritime Field Data Exchange Based on ISO 19847/19848," *Journal of the Korea Institute of Information and Communication Engineering*, Vol. 22, pp. 1577-1588, 2018.

[10] B.S Kim and T.G. Kim, "Development Process of Distributed Systems using System Entity Structure Formalism," *Proceeding of the Summer Conference of the Institude of Electronics and Information Engineers*, Vol. 35, No. 1, pp. 1997-2000, 2012.

[11] Y.S. Han, "A Study on the Curriculum Design Engine Using a SES/DEVS," *Journal of Engineering Education Research*, Vol. 16, No. 5, pp. 18-23, 2013.

[12] A. Brandsæter and O.L. Osen, "Assessing Autonomous Ship Navigation Using Bridge Simulators Enhanced by Cycle-Consistent Adversarial Networksm," *Proceedings of the Institution of Mechanical Engineers, Part O: Journal of Risk and Reliability*, 2021.

[13] S.K. Katsikas, "Cyber Security of the Autonomous Ship," *Proceedings of the 3rd ACM Workshop on Cyber-Physical System Security,* pp. 55-56, 2017.

[14] M.J. Blas, H. Leone, and S. Gonnet, "DEVS-Based Formalism for the Modeling of Routing Processes," *Software and Systems Modeling*, Vol. 21, pp. 1179-1208, 2022.

[15] S. Cheon, D. Kim, and B.P. Zeigler, "System Entity Structure for XML Meta Data Modeling; Application to the US Climate Normals," *Subcommittee on Security and Defence*, pp. 216-221, 2008.

[16] Y. Han, "Effectiveness Analysis of Programming Education for College of Education Student Based on Information Processing Theory Applied DEVS Methodology," *Journal of Korea Multimedia Society*, Vol. 23, No. 9, pp. 1191-1200, 2020.

[17] B.P. Zeigler, A. Muzy, and E. Kofman, *Theory of Modeling and Simulation: Discrete Event & Iterative System Computational Foundations*, *Academic Press*, 2018.

[18] Common Vulnerabilities and Exposures (CVE-2022-0847), https://cve.mitre.org/cgi-bin/cvename.cgi?name=CVE-2022-0847 (Accessed May 10, 2022).

이 창 의

2009년 한국해양대학교 학사 (컴퓨터 공학)
2020년 한국해양대학교 대학원 석사(컴퓨터 공학)
2002년 한국해양대학교 대학원 박사수료(컴퓨터 공학)

현재 한국건설생활환경시험연구원(KCL) 선임연구원
관심분야 : 소프트웨어 품질, 소프트웨어 기능안전 및 위험성 평가

이 서 정

1989년 숙명여자대학교 학사 (전산학)
1991년 숙명여자대학교 대학원 석사(전산학)
1998년 숙명여자대학교 대학원 박사 (전산학)

현재 한국해양대학교 기관시스템공학부 교수
관심분야 : 해양소프트웨어품질, 소프트웨어 기능안전, 차세대 전자해도표준