Journal of The Korea Institute of Information Security & Cryptology ISSN 1598-3986(Print)
VOL.31, NO.4, Aug. 2021 ISSN 2288-2715(Online)
https://doi.org/10.13089/JKIISC.2021.31.4.617

IoT 인프라 공격 확산 방지 기술 성능 검증을 위한 악성코드 고속 확산 기법 연구*

황 송 이,[1†] 김 정 녀[2‡]
[1]과학기술연합대학원대학교 (대학원생), [2]한국전자통신연구원 (연구원)

A Study on the High-Speed Malware Propagation Method for Verification of Threat Propagation Prevent Technology in IoT Infrastructure*

Song-yi Hwang,[1†] Jeong-Nyeo Kim[2‡]
[1]University of Science and Technology (Graduate student),
[2]Electronics and Telecommunications Research Institute (Researcher)

요 약

네트워크에 연결된 사물인터넷(Internet of Things, IoT) 기기는 보안 솔루션이 적용되지 않아 ICT(Information & Communications Technology) 인프라의 심각한 보안 위협으로 전락했다. 더군다나 IoT 기기의 특성상 자원제약이 많아 기존의 보안 솔루션을 적용하기 어렵다. 그 결과 사물인터넷 기기는 사이버 공격자의 공격 대상이 됐으며, 실제로도 사물인터넷 기기를 대상으로 한 악성코드 공격이 해마다 꾸준히 증가하고 있다. 이에 IoT 인프라를 보호하기 위해 여러 보안 솔루션이 개발되고 있지만, 기능이 검증되지 않은 보안 솔루션을 실제 환경에 적용하기엔 큰 위험이 따른다. 따라서 보안 솔루션의 기능과 성능을 검증할 검증 도구도 필요하다. 보안 솔루션이 다양한 보안 위협에 대응하는 방법도 다양하므로, 각 보안 솔루션의 특징을 기반으로 한 최적의 검증 도구가 필요하다. 본 논문에서는 IoT 인프라에 빠른 속도로 악성코드를 전파하는 악성코드 고속 확산 도구를 제안한다. 또한, IoT 인프라에서 확산하는 공격을 빠르게 탐지하고 차단하는 보안 솔루션의 기능과 성능을 검증하기 위해 개발된 악성코드 고속 확산 도구를 이용한다.

ABSTRACT

Internet of Things (IoT) devices connected to the network without appropriate security solutions have become a serious security threat to ICT infrastructure. Moreover, due to the nature of IoT devices, it is difficult to apply currently existing security solutions. As a result, IoT devices have easily become targets for cyber attackers, and malware attacks on IoT devices are actually increasing every year. Even though several security solutions are being developed to protect IoT infrastructure, there is a great risk to apply unverified security solutions to real-world environments. Therefore, verification tools to verify the functionality and performance of the developed security solutions are also needed. Furthermore, just as security threats vary, there are several security solution s that defend against them, requiring suitable verification tools based on the characteristics of each security solution. In this paper, we propose an high-speed malware propagation tool that spreads malware at high speed in the IoT infrastructure. Also, we can verify the functionality of the security solution that detect and quickly block attacks spreading in IoT infrastructure by using the high-speed malware propagation tool.

Keywords: IoT Malware, Propagation, Malware attack tool, Verification

Received(04. 16. 2021), Modified(06. 10. 2021), Accepted(06. 11. 2021)

* 이 논문은 2021년도 정부(과학기술정보통신부)의 재원으로 정보통신기획평가원의 지원을 받아 수행된 연구임 (No. 2018-0-00231, (IoT 2세부) IoT 인프라 공격 확산 방어를 위한 상황 적응형 보안 자율제어 기술개발)

† 주저자, syhwang@ust.ac.kr
‡ 교신저자, jnkim@etri.re.kr(Corresponding author)

I. 서 론

사물인터넷(Internet of Things, IoT)은 가전, 의료, 제조 및 생산 등 많은 산업에 적용되었다[1]. 그 결과 서로 다른 하드웨어와 OS, 서로 다른 통신 프로토콜과 표준, 그리고 다양한 애플리케이션 소프트웨어가 만들어졌다[2]. 이러한 통일성 없는 거대한 사물인터넷 프레임워크는 새롭고 다양한 보안 문제를 초래하였다. 공격자는 취약점이 있는 대량의 IoT 기기를 노려 악성코드 공격을 시도했고, 대규모 사이버 범죄를 유발했다. 실제로 2016년도에 도메인 서비스 업체인 다인(Dyn)이 미라이(mirai) 봇넷으로부터 직접적인 분산 서비스 거부(Distributed Denial of Service, DDoS) 공격을 받았고, 이 공격으로 인해 Twitter, BBC, CNN, Netflix 등 유명 웹 사이트의 접속 장애로 이어졌다. 공장 출하 시 설정해놓은 네트워크 접속 정보를 그대로 사용했던 120만 대 이상의 IoT 기기가 미라이 악성코드에 감염됐다[3]. 거대한 미라이 봇넷은 다인(Dyn)을 목표로 DDoS 공격을 퍼부었고 대규모 인터넷 정전 사태를 일으켰다. 특히 미라이 악성코드의 소스 코드가 인터넷에 공개된 이후로 사이버 공격자들은 IoT 기기를 감염시키기 위해 지금까지도 수많은 미라이 변종 악성코드를 생성 및 배포하고 있다[4].

실제로 2019년도 기준, 하루에 1,200건씩 총 44만 건의 IoT 악성코드가 발견됐다[5]. 또한, IoT 악성코드 공격은 2020년에 3,430만 건을 기록하여 2019년보다 5% 증가했다[6]. IoT 기기가 사이버 공격자의 표적이 되는 이유는 IoT 기기의 보안 수준이 여전히 매우 낮기 때문이다. 현재의 IoT 장치는 오래된 공격 기법에 취약할 뿐만 아니라 특정 IoT 기기를 겨냥한 IoT 악성코드 공격과 같은 새로운 공격에 쉽게 보안 되지 않는다. 가장 불행히도, 모든 IoT 장치에서 내보낸 트래픽의 98%가 암호화되지 않아 네트워크상의 개인 및 기밀 데이터가 노출된다[7].

2020년에는 117억 개의 IoT 기기가 인터넷에 연결됐다. 전 세계 217억 개의 활성 커넥터 중 54%를 차지한다[8]. 2025년까지 300억 개 이상의 IoT 기기가 인터넷에 연결될 것으로 예상된다[9,10]. 이런 추세에 따라 IoT 기기의 보안성이 여전히 낮은 상태에서 네트워크에 연결되는 IoT 기기가 늘어날 경우, IoT 기기를 겨냥한 사이버 공격도 늘어날 것으로 예상돼 피해가 클 것으로 보인다.

보안 위협이 다양하듯이 이를 방어하는 방법도 다양하다. 공격 피해를 줄이고자 침입 탐지, 암호화 알고리즘, DDoS 공격 탐지, 하드웨어 기반 격리 등 다양한 보안 솔루션이 보안 IoT 인프라를 위해 제안되었다[11]. 다만 아무리 좋은 보안 솔루션을 개발해도 기능과 성능이 검증되지 않는 한 현실 세계에 적용하기 어렵다. 무작위로 발생하는 위협을 보안 솔루션이 막을 수 있다는 확신이 없기 때문이다. 따라서 실제 공격과 똑같은 위협을 발생시킬 수 있는 검증 도구로 보안 솔루션의 기능과 성능을 테스트하여 보안 솔루션에 대한 신뢰성과 보안성능을 모두 입증할 필요가 있다. 이를 위해 각 보안 솔루션이 제안하는 보안 기능에 따른 적절한 검증 도구가 필요하다.

따라서 본 논문에서는 IoT 인프라 보호를 위한 공격 확산 방지 기술 중 스마트 세그멘테이션 솔루션(SSS)의 기능과 성능을 검증하기 위한 검증 도구를 제안한다. IoT 인프라에 확산하는 위협을 빠르게 탐지하고 차단하는 SSS 보안 솔루션의 검증을 위해 악성코드 감염 및 확산을 위협 확산이라고 간주하여 IoT 환경에서 고속으로 악성코드를 확산시키는 악성코드 고속 확산 도구를 구현한다. 악성코드 고속 확산 도구는 SSS 보안 솔루션뿐만 아니더라도 IoT 인프라 내 악성코드 공격을 탐지하고 차단하는 보안 솔루션의 기능을 검증하는 데 적합하다. 2장에서는 악성코드 고속 확산 도구의 기반이 된 IoT 인프라 보안 솔루션인 공격 확산 방지 기술과 미라이 악성코드를 소개하고, 이어 3장에서는 악성코드 고속 확산 도구와 관련된 연구를 소개한다. 이어서 4장에서는 IoT 인프라 위협 방지 보안 솔루션의 기능 검증을 위한 악성코드 고속 확산 도구의 확산 기법에 관해 설명하고, IoT 인프라 위협 방지 보안 솔루션의 성능 검증을 위하여 확산 속도에 영향을 주는 요인을 분석한다. 그런 다음 5장에서는 스캔 기법, 스캔 동작을 수행하는 봇의 개수, 1회 포트 스캔 시 내보내는 패킷 개수에 따라 악성코드 확산 실험을 수행하고, IoT 인프라의 전체 기기가 감염되는 데 걸린 시간을 비교한다. 마지막으로, 6장에서는 논문을 마무리하고 추후 연구에 대한 방향을 제시한다.

II. 배 경

2.1 IoT 인프라 보호를 위한 공격 확산 방지 기술

최근 여러 산업에서 IoT 수요가 늘면서 IoT 기기의 수요가 증가하고 있다[12,13]. 이에 따라 IoT

기기들이 대규모로 구성된 초연결 IoT 인프라로 확장되고 있다. 하지만 IoT 기기의 수가 증가한 만큼 IoT 인프라 보안 사고 또한 증가했다[14]. 보안 수준이 낮은 IoT 기기 하나하나가 IoT 인프라의 잠재적인 보안 위협점이 되어 IoT 악성코드 공격, 서비스 거부 공격, 개인 및 기밀 정보 유출 등과 같은 보안 사고를 초래했기 때문이다.

사물인터넷 환경에서 발생할 수 있는 보안 사고를 방지하고 방어하기 위해 여러 IoT 인프라 보호 기술들이 연구되고 있다. 그중에서도 본 연구실에서는 위협 확산 방지 기술을 개발하였는데, 주요 기술은 크게 3가지로 스마트 세그멘테이션 솔루션, 동적 경계망 솔루션, 원격검증 솔루션으로 나누어져 있다. 공격 확산 방지 기술의 개념도는 아래 그림 1과 같다.

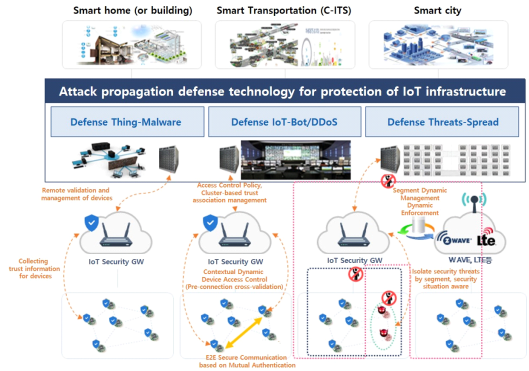

Fig. 1. Conceptual diagram of technology to prevent malware propagation attacks for IoT infrastructure security

2.1.1 스마트 세그멘테이션 솔루션 (Smart Segmentation Solution)

확산 방지 기술 중 스마트 세그멘테이션 솔루션(SSS)은 IoT 인프라에서 빠르게 확산하고 있는 위협을 탐지하고 차단하는 기술이다. SSS 보안 솔루션의 보안 관리 시스템으로 IoT 인프라 내의 게이트웨이를 관리함으로써 게이트웨이에 연결된 여러 대의 IoT 기기들을 통제한다. 인프라를 구성하는 모든 제품의 위험도를 계산하여 위험 수준 별로 기기/네트워크(게이트웨이 등)/응용서비스 단위로 네트워크에서 격리한다. IoT 인프라를 구성하는 구성 요소의 네트워크를 자율적으로 제어함으로써 보안 위협 확산으로 인한 IoT 서비스 피해를 최소화하고 안전한 인프라 환경을 구축한다.

2.1.2 동적 경계망 솔루션 (Dynamic Perimeter Solution)

동적 경계망 솔루션(DPS)은 분산 서비스 거부(DDoS) 공격을 방지하는 기술이다. DPS 보안 솔루션은 신뢰 연결(trusted channel)을 관리하는 제어 서버를 IoT 인프라 입구에 두어 IoT 인프라에 기기가 접속하고자 할 때마다 DPS 제어 서버를 통하도록 한다. DPS 제어 서버를 통해 미리 인가된 기기만이 접속될 수 있도록 제어하여 오용 및 도용을 할 수 있는 악의적인 단말은 접속할 수 없도록 한다. DPS 제어 서버를 한 번 거쳐 사전에 인가받은 기기만이 IoT 인프라에 접속할 수 있어 IoT 인프라를 대상으로 한 분산 서비스 거부 공격이 예방된다.

2.1.3 원격검증 솔루션 (Remote Attestation Solution)

확산 방지 기술 중 마지막 솔루션인 원격검증 솔루션(RAS)은 IoT 기기의 악성코드 감염 여부를 탐지하는 기술이다. 악성코드를 탐지하고 감염된 기기를 치료하기 위해 백신을 사용해도 되지만, 하드웨어 성능에 제약이 있는 IoT 기기에서는 백신을 사용하기 적합하지 않다. 왜냐하면 IoT 기기를 대상으로 한 악성코드는 계속해서 증가하고 있으며, 엄청난 수의 악성코드 패턴을 처리하기에는 IoT 기기의 하드웨어 사양이 좋지 않다. 그래서 RAS 보안 솔루션은 펌웨어, OS 커널 등의 무결성 검증을 통하여 악성코드 감염 여부를 확인한다.

2.2 미라이 악성코드

미라이 봇넷 인프라는 그림 2와 같이 미라이 봇과 미라이 봇을 관리하고 제어하는 C&C 서버, 감염시킬 기기의 정보를 수집하는 리포트 서버, C&C 서버와 리포트 서버의 공인 IP 주소를 알려주는 DNS 서버 그리고 악성코드를 IoT 기기에 주입하는 악성코드 로더로 구성된다.

미라이 봇넷이 수행하는 동작은 크게 2가지가 있다. 첫째는 취약한 IoT 기기를 찾아 감염시키는 것이고, 둘째는 공격 대상에 DDoS 공격을 하는 것이다. 취약한 IoT 기기를 많이 감염시킬수록 미라이 봇넷의 규모가 커지고 그만큼 거대한 DDoS 공격을 일으킬 수 있어서 미라이 악성코드는 전 세계에 놓인

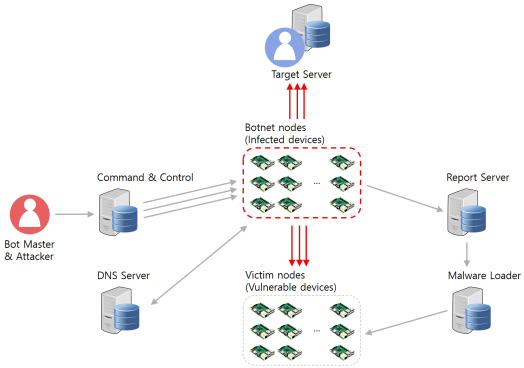

Fig. 2. Mirai botnet configuration diagram

취약한 IoT 기기들을 최대한 많이 찾고자 했다. 이를 위해 미라이 악성코드가 적용한 확산 방식은 랜덤 스캔 기법과 로더 배포 확산 기법이다.

2.2.1 랜덤 스캔 기법

미라이 악성코드는 봇넷의 규모가 커질수록 더 강력한 DDoS 공격을 일으킬 수 있었기 때문에 최대한 많은 IoT 기기를 감염시키고자 하였다. 그래서 미라이 악성코드는 장치에는 할당할 수 없는 표 1의 특정 IP 주소를 제외한 모든 IP 주소를 스캔하기 위해 랜덤 스캐닝 방식을 적용했다.

Table 1. Blacklist IP addresses

Reason	IP Range
Loopback	127.0.0.0/8
Invalid address space	0.0.0.0/8
General Electric Company	3.0.0.0/8
Hewlett-Packard Company	15.0.0.0/7
US Postal Service	56.0.0.0/8
Internal network	10.0.0.0/8
	192.168.0.0/16
	172.16.0.0/14
IANA NAT reserved	100.64.0.0/10
	169.254.0.0/16
IANA Special use	198.18.0.0/15
Multicast	224.*.*.*+

2.2.2 로더 배포 악성코드 확산 기법 절차

미라이 악성코드에서는 봇(bot)과 멀웨어 로더(malware loader)의 역할이 정확히 구분된다. 미라이 봇은 취약한 기기를 찾고, 멀웨어 로더는 취약한 기기에 악성코드를 주입한다. 미라이 악성코드는 IoT 기기를 감염시키기 위해 그림 3과 같이 총 6단계의 확산 단계를 거치며, 미라이 봇은 1~3단계를 수행하고 멀웨어 로더는 4~6단계를 수행한다.

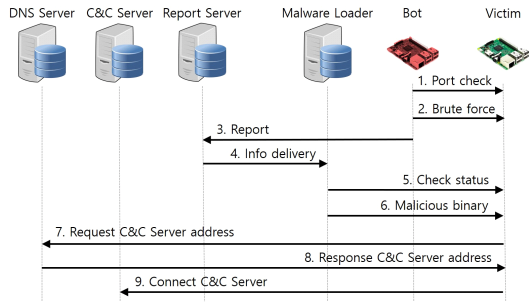

Fig. 3. Mirai malware's propagation process

1단계. 텔넷 포트 스캔. 미라이 봇은 텔넷 서비스를 사용하는 기기를 스캔한다. 봇은 목적지 포트 번호를 23 또는 2323으로 설정하고 목적지 IP 주소를 임의로 설정한 패킷을 보낸 다음 응답을 확인하여 기기에서 텔넷 서비스가 활성화되었는지 확인한다. 미라이 악성코드는 TCP Half(SYN) Open 스캔 방식을 사용해서 취약한 기기(victim)와 TCP 연결을 맺지 않는다. 그러므로 취약한 기기에 로그가 남지 않아 스텔스 스캔이 가능하다.

2단계. 텔넷 로그인을 위한 자격증명 무차별 대입 공격. 미라이 봇은 텔넷 서비스가 활성화된 기기를 대상으로 접속을 시도한다. 미라이 악성코드에 저장된 62개의 자격증명으로 최대 30초 동안 무차별 대입 공격을 한다.

3단계. 취약한 기기의 정보 전달. 텔넷 로그인이 성공하면 미라이 봇은 취약한 기기의 IP 주소와 포트 번호, 텔넷 서비스 로그인 계정을 리포트 서버에 전달한다. 그래서 리포트 서버는 미라이 봇이 넘겨주는 취약한 기기의 정보를 받기 위해 48101 포트를 열어둔다.

4단계. 멀웨어 로더에 정보 전달. 리포트 서버는 미라이 봇으로부터 전달받은 취약한 기기의 정보를 전달받은 순서대로 멀웨어 로더에게 넘겨준다.

5단계. 취약한 기기의 상태 확인. 취약한 기기에 미라이 악성코드를 주입하기 위한 사전 조사가 이뤄진다. 멀웨어 로더는 리포트 서버로부터 전달받은 정보로 취약한 기기에 접속한다. 그 뒤, 취약한 기기에

대한 CPU 아키텍처, 쉘(shell)의 종류, 비지박스(busybox) 설치 여부, wget/tftp 설치 여부 등 악성코드를 주입하는 데 필요한 정보를 수집한다.

6단계. 미라이 악성코드 주입. 멀웨어 로더는 취약한 기기에 미라이 악성코드를 주입한다. 멀웨어 로더는 수집한 정보를 기반으로 미라이 악성코드를 주입하는 데 사용할 적절한 프로그램과 CPU 아키텍처에 맞는 적합한 미라이 악성코드 바이너리를 선택한다.

III. 관련 연구

3.1 IoT 악성코드

이 장에서는 IoT 악성코드로 대표적인 배쉬라이트, 하지메, 브리커, IoT 리퍼, VPNFilter에 대해 분석하였다[15].

3.1.1 배쉬라이트(BASHLITE)

미라이 악성코드와 동일하게 ARM과 MIPS 아키텍처 기반 리눅스 IoT 기기를 감염시켜 DDoS 공격에 사용되었다. 배쉬라이트의 소스 코드가 일부 공개되어 배쉬라이트 변형 악성코드가 많이 만들어졌고, 여러 배쉬라이트 변형 악성코드 중에 미라이 악성코드도 포함되어 있다. 배쉬라이트가 취약한 기기를 스캔한 기법은 **랜덤 스캔 기법**이며, 취약한 기기를 악성코드에 감염시키기 위해 **하나의 CVE 취약점 공격** 또는 **비밀번호 사전 공격**하였다. 배쉬라이트는 미라이와 동일하게 **중앙집중식 구조**로 C&C 서버에서 봇넷을 관리한다. 배쉬라이트는 다른 악성코드가 취약한 기기를 감염시킬 수 없도록 공격 대상으로 사용된 포트를 종료하는 특징을 가진다.

3.1.2 하지메(Hajime)

하지메는 특정 유형 기기만을 대상으로 공격하지 않고 인터넷에 연결된 ARM, MIPS, x86/64 아키텍처 기반의 모든 기기를 공격한다. 그러나 디지털 비디오 레코더(DVR), 웹카메라, 라우터에서 주로 하지메 악성코드를 발견할 수 있었다. 하지메는 현재까지 어떠한 악의적인 행동도 보이지 않았지만, 하지메 봇넷을 DDoS 공격에 사용할 것으로 예상한다. 하지메는 취약한 기기를 스캔한 기법은 **랜덤 스캔 기법**이며, 취약한 기기를 악성코드에 감염시키기 위해 **비밀번호 사전 공격**하였다. 하지메는 배쉬라이트, 미라이와 다르게 **P2P(Peer to Peer) 구조**로 C&C 서버가 없다. 하지메는 포트를 종료하는 특징 외에도 악성코드 바이너리를 실행과 동시에 제거하며, 재부팅을 방지하며, 취약한 기기의 아키텍처를 스캔하고, 악성코드 프로세스가 정상적인 프로세스인 척 위장한다. 또한, 자신 외에 취약한 기기에서 동작하는 봇을 제거하고, 코드의 모듈화 또는 업데이트 시스템을 가진다.

3.1.3 브리커(Bricker)

브리커도 미라이 악성코드와 동일하게 리눅스 비지박스 기반 IoT 장치를 감염시킨다. 다만, 브리커는 악성코드에 감염시킨 봇을 DDoS 공격에 사용하지 않고, 악성코드에 감염된 취약한 IoT 기기를 기능 불능 상태로 만든다. 브리커는 미라이와 동일하게 취약한 기기를 스캔한 기법은 **상태를 저장하지 않는 랜덤 스캔 기법**을 적용하였으며, 취약한 기기를 악성코드에 감염시키기 위해 **비밀번호 사전 공격**하였으나 비밀번호마다 **가중치**를 두어 자격증명이 선택될 확률은 가중치 값에 따랐다. 브리커 역시 미라이와 동일하게 **중앙집중식 구조**를 갖는다.

3.1.4 IoT 리퍼(IoT Reaper)

IoT 리퍼는 특정 제조업체의 IoT 기기를 겨냥한 악성코드이다. IoT 리퍼에 감염된 봇은 DDoS 공격에 사용되었다. IoT 리퍼는 취약한 특정 기기를 스캔한 기법은 **랜덤 스캔 기법**이며, 취약한 기기를 악성코드에 감염시키기 위해 원격코드 실행 취약점, 웹 인증 우회 취약점과 같은 **다양한 CVE 취약점을** 공격 벡터로 갖는다. 또한, IoT 리퍼도 미라이와 동일하게 **중앙집중식 구조**이다. IoT 리퍼는 하지메와 동일하게 하지메가 갖는 7가지 특성을 모두 갖는다.

3.1.5 VPNFilter

VPNFilter는 ARM, MIPS, x86/64 아키텍처 기반의 기기와 특정 제조업체 기기를 감염시켜 PDoS, 방화벽 DoS, 데이터 유출, 중간자 공격 등 다양한 공격을 일으킨다. VPNFilter가 취약한 기기를 스캔하기 위해 상태를 저장하지 않는 **랜덤 스캔 기법**을 적용하였으며, 취약한 기기를 악성코드에 감염시

키기 위해 **다양한 CVE 취약점**을 공격하였다. VPNFilter 또한 미라이와 동일한 **중앙집중식 구조**이다. VPNFilter는 취약한 기기의 펌웨어를 변경하므로, 취약한 기기가 재부팅되어도 봇이 계속 유지되는 특징이 있다. 다른 IoT 악성코드의 특징과 비슷하게 코드의 모듈화 및 업데이트 시스템을 가지며, 재부팅을 방지하고, 취약한 기기의 아키텍처를 스캔한다. 또한, VPNFilter는 C&C 서버 탐지를 회피하는 목적으로 도메인 주소를 무작위로 생성하는 DGA 알고리즘을 사용한다.

분석 결과 IoT 악성코드가 취약한 기기를 스캔하는 기법과 취약한 기기에 악성코드를 감염시키는 기법, 그리고 봇넷의 구성 요소와 구조는 거의 일치하였다. 그중에서도 미라이 악성코드는 코드 모듈화 및 업데이트 시스템과 악성코드가 유지되는 특성을 제외하면 다섯 가지 IoT 악성코드가 가진 모든 특징을 다 포함하였다. 더욱이, 수많은 IoT 악성코드 중 유일하게 미라이 악성코드만 인터넷에 소스 코드가 공개되었다. 따라서 본 논문에서 SSS 보안 솔루션 검증을 위해 구현한 악성코드 고속 확산 도구 또한 미라이 악성코드를 기반으로 구현하였다.

IV. 스마트 세그멘테이션 솔루션 검증을 위한 악성코드 고속 확산 도구

본 연구에서 제안하는 악성코드 고속 확산 도구는 특정 인프라 또는 내부 네트워크 환경에 악성코드를 빠르게 확산시키는 검증 도구이다. 악성코드 고속 확산 도구는 IoT 인프라 내 네트워크 위협을 빠르게 탐지하고, 탐지된 위협을 차단하는 네트워크 위협 방지 기술을 검증한다. 특히, 본 연구실에서 개발한 공격 확산 방지 기술 중의 하나인 스마트 세그멘테이션 솔루션(SSS)의 기능을 검증하기에 적합한 검증 도구이다. 악성코드 고속 확산 도구는 IoT 인프라에 악성코드를 확산시켜 SSS 보안 솔루션이 확산하는 위협을 빠르게 탐지하고 차단하는지 검증하기 위하여 IoT 보안의 심각성을 알린 미라이(mirai) 악성코드를 기반으로 구현하였다. 그러나 미라이 악성코드의 목적은 전 세계에 놓인 취약한 IoT 기기에 악성코드를 감염시키고, 거대한 DDoS 공격을 일으키는 목적으로 개발되었기 때문에 미라이 악성코드의 랜덤 스캔 방식과 로더 배포 확산 방식은 성능 검증용으로 사용하기에 적합하지 않다.

4.1 미라이 악성코드를 검증 도구로 사용 시 발생하는 문제점

미라이 악성코드의 목적은 악성코드 고속 확산 도구의 목적과 상이해 미라이 악성코드의 랜덤 스캔 기법과 로더 배포 악성코드 확산 기법을 검증 도구에 적용할 수 없다. 미라이 악성코드의 스캔 기법과 확산 기법을 검증 도구에 적용하면 다섯 가지 문제가 존재한다.

4.1.1 광범위한 스캔 범위로 인한 특정 대상 인프라 내 확산 어려움

미라이 악성코드가 사용한 랜덤 스캔 기법은 스캔할 IP 주소 범위가 제한된 특정 인프라를 대상으로 악성코드 공격 시 적합하지 않다. 랜덤 스캔 기법은 전 세계적으로 할당된 IPv4 주소 중 무작위로 한 IP 주소를 선택해 해당 IP 주소를 할당받은 기기를 스캔한다. 현재 전 세계에 36억 개 이상의 IPv4 주소가 할당되어 있으므로[16], 특정 인프라 내 IoT 기기가 스캔 될 확률은 36억 분의 해당 인프라 내 기기의 개수이다. 따라서, 미라이가 적용한 랜덤 스캔 기법은 특정 인프라 내에 악성코드를 빠르게 확산하기에 적합하지 않다.

4.1.2 멀웨어 로더에서 내부 네트워크 접근 불가

미라이 악성코드는 내부 네트워크를 스캔하지 않는다. 멀웨어 로더가 취약한 기기에 악성코드 바이너리를 주입하기 위해 취약한 기기와 네트워크 통신을 맺

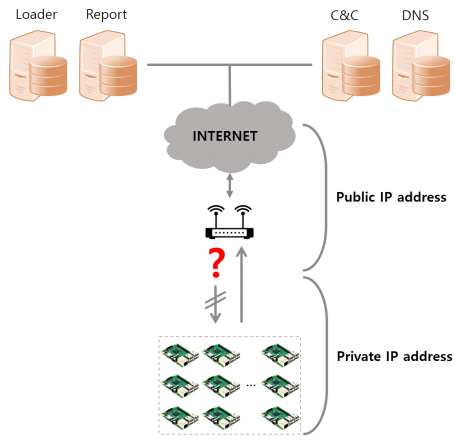

Fig. 4. Inaccessible to internal network device

어야 하지만, 그림 4와 같이 사설 IP 주소를 할당받은 기기에는 멀웨어 로더가 접근할 수 없기 때문이다. 그래서 미라이 악성코드는 루프백, 내부 네트워크 및 멀티캐스트 주소를 제외한 공인 IP 주소를 할당받은 장치에 대해서만 악성코드 공격을 시도한다.

4.1.3 스캔 대상 범위를 고려하지 않은 스캔 패킷 개수로 인한 중복 감염 증가

특정 대상 인프라에서 악성코드 공격 수행 시 인프라 내 스캔 대상이 되는 IoT 기기 개수는 적지만, 그에 반해 스캔 패킷 개수가 많으면 여러 개의 봇이 동시에 하나의 기기를 대상으로 확산 과정을 수행할 가능성이 크다. 미라이 악성코드는 스캔하는 IP 주소 범위에 제한이 없으며 임의로 IP 주소를 선택하기 때문에 여러 개의 봇이 같은 기기를 감염시킬 가능성은 거의 없다. 그러나 사용 가능한 IP 주소 범위가 한정된 내부 네트워크에서 봇당 160개 스캔 패킷을 내보내면 여러 개의 봇이 동일한 기기를 중복으로 스캔할 가능성이 크다.

또한, 취약한 기기가 악성코드에 감염되어 봇이 되기 위해선 취약한 장치에서 악성 프로그램이 실행되어야만 한다. 만일 취약한 기기가 이미 악성 프로그램을 내려받았더라도 악성 프로그램이 실행되기 전이라 봇이 되지 못했다면 다른 봇이 해당 기기를 스캔할 수 있다. 그림 5와 같이 취약한 기기가 악성코드를 내려받는 시간과 내려받은 악성코드를 실행하는 시간 사이에는 공백이 있다. 취약한 장치는 이미 악성 프로그램을 한 번 내려받았더라도 악성 프로그램이 실행되어 텔넷 서비스를 종료하기 전이라면 동일한 악성 프로그램을 여러 번 내려받는 중복 감염이 발생할 가능성이 크다. 취약한 기기가 악성코드에 중복으로 감염되면

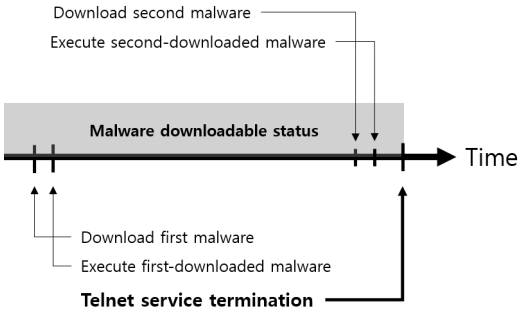

Fig. 5. Time line of malware download and execution

시간 공백 사이에 여러 번 내려받은 악성코드가 차례대로 실행되고, 자신 외에 다른 악성코드를 탐지해 종료시키는 킬러 프로세스에 의해 서로를 종료시키고, 결과적으로 악성코드가 모두 종료되어 취약한 기기를 봇으로 만들 수 없다.

4.1.4 인프라 내 모든 봇 스캔 동작 시 과도한 스캔 패킷으로 인한 인프라 네트워크 장비 과부하

특정 인프라를 공격 대상으로 악성코드 공격 시 해당 인프라 내 모든 봇이 스캔 동작을 수행하면 네트워크 정체 트래픽이 증가하여 네트워크 장비에 과부하가 발생한다. 이는 포트 스캔을 위해 봇이 전송하는 모든 스캔 패킷은 특정 인프라의 네트워크 장비를 거치기 때문이다. 특정 인프라 내 스캔 동작을 수행하는 봇의 개수가 적으면 네트워크 장비는 봇이 내보내는 스캔 패킷을 지연 없이 처리한다. 그러나 스캔 동작을 수행하는 봇의 개수가 증가하면 네트워크 장비가 처리해야 하는 스캔 패킷의 양이 증가하여 네트워크 장비가 처리할 수 있는 트래픽을 넘어서게 된다. 이러한 이유로 특정 인프라 내 스캔 동작을 수행하는 봇의 개수가 많고, 한 봇당 포트 스캔을 위해 내보내는 스캔 패킷 개수가 많은 경우 네트워크 장비는 정상적으로 동작하지 않는다.

미라이 악성코드는 취약한 기기를 찾기 위해 포트 스캔 1회당 160개의 패킷을 전송하며, 포트 스캔 횟수에는 제한이 없다. 이처럼 미라이 악성코드와 동일하게 포트 스캔 1회당 내보내는 패킷의 개수를 160개로 설정하면, 봇의 개수가 증가할수록 네트워크 장비는 160배수로 증가하는 패킷을 처리해야 한다. 이는 특정 인프라 내 IoT 기기를 대상으로 악성코드를 빠르게 확산하는 검증 도구의 목표와 다르게 인프라 내 네트워크 장비를 대상으로 한 DDoS 공격으로 변질할 수 있다. 이 경우 네트워크 장비의 성능 저하로 악성코드 확산 속도가 느려지거나, 네트워크 장비가 정상적인 동작을 수행하지 못해 악성코드를 확산할 수 없거나, 보안 솔루션에 의해 공격이 탐지될 확률이 높다.

4.1.5 인프라 구축의 비효율성

사설 IP 주소를 사용하는 내부 네트워크 내에서 악성코드를 전파하려면 C&C 서버, DNS 서버, 리포트 서버, 멀웨어 로더와 같은 봇넷 인프라가 내부 네트워크에 구성되어 있어야 한다. 하지만 이 방법은 공격

대상이 되는 인프라마다 봇넷 인프라를 구축해야 하는 비효율적인 문제를 갖는다.

또한, 내부 네트워크 통신은 데이터 링크 계층 헤더의 MAC 주소만 참조하기 때문에 네트워크 계층 헤더의 IP 주소를 참조하지 않는다. 내부 네트워크에 봇넷 인프라를 구축한 뒤 악성코드 확산을 수행하는 경우 네트워크 계층 트래픽을 분석하여 보안 위협을 탐지하는 보안 솔루션이 위협을 탐지할 수 없다. 결국 네트워크 위협 방지 보안 솔루션의 기능 및 성능을 검증할 수 없는 문제가 발생한다.

4.2 SSS 기능 검증을 위한 문제점 해결방안

미라이 악성코드의 랜덤 스캔 기법과 로더 배포 확산 기법을 검증 도구에 적용하면 발생하는 문제를 해결하기 위해 제한 범위 스캔 기법과 D2D 커맨드 인젝션 확산 기법을 적용한 악성코드 고속 확산 도구를 제안한다. 제한 범위 스캔 기법과 D2D 커맨드 인젝션 확산 기법을 악성코드 고속 확산 도구에 적용한 결과, **내부 네트워크 내에 사설 IP를 할당받은 IoT 기기에도 악성코드 확산이 가능**하고, **인프라 구축의 비효율성을 해결**하며, **특정 대상 인프라 내 빠른 악성코드 확산이 가능**해져 IoT 인프라에 확산하는 위협을 빠르게 탐지하고 차단하는 SSS 보안 솔루션의 검증이 가능하다.

4.2.1 제한 범위 스캔 기법

제한 범위 스캔 기법은 사설 IP 주소를 할당받은 네트워크 영역을 스캔하는 기법이다. 제한 범위 스캔 기법으로는 순차 스캔 방식, 분할 순차 스캔 방식, 제한 랜덤 스캔 방식, 그리고 분할 랜덤 스캔 방식이 있다. 네 가지 스캔 방식을 통해 인프라의 전체 IoT 장치로 악성코드가 퍼지는 시간을 확인한다. 이 실험 결과를 통해 내부 네트워크와 같이 스캔할 IP 주소의 범위가 제한된 조건에서 악성코드를 빠르게 전파할 수 있는 최적의 스캔 방식을 찾는다.

4.2.1.1 순차 스캔 방식

순차 스캔 방식은 그림 6과 같이 제한된 IP 주소 범위의 처음부터 끝까지 순차적으로 스캔한다. 순차 스캔 방식에서는 스캔 범위를 구하기 위해 봇의 로컬 네트워크 IP 주소와 서브넷 마스크가 필요하다. 봇

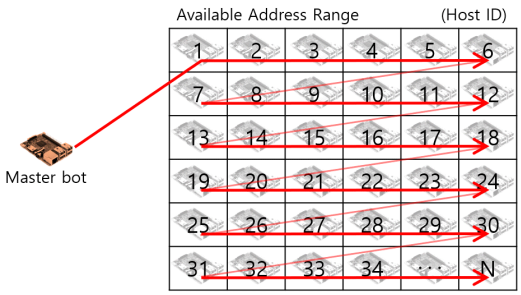

Fig. 6. Sequential scanning method

의 로컬 네트워크 IP 주소에서 네트워크 ID와 호스트 ID를 구분하고 사용 가능 주소 범위를 구한다. 사용 가능 주소 범위는 기기가 할당받을 수 있는 내부 네트워크의 IP 주소 범위로 봇이 스캔할 제한된 IP 주소 범위를 의미한다.

예를 들어 봇의 로컬 네트워크 IP 주소가 192.168.1.101/24인 경우 네트워크 ID는 192.168.1.0이고 호스트 ID는 101이다. 봇이 포함된 사설 네트워크의 사용 가능 주소는 192.168.1.1에서 192.168.1.254까지다. 봇은 사용 가능 주소의 처음부터 끝까지 차례대로 스캔한다. 만약 봇이 스캔하는 IP 주소가 사용 가능 주소 범위를 벗어나는 경우 다시 사용 가능 주소 범위의 처음으로 돌아간다.

4.2.1.2 분할 순차 스캔 방식

분할 순차 스캔 방식은 순차 스캔 방식과 동일하게 제한된 IP 주소 범위의 처음부터 끝까지 순차적으로 스캔한다. 다만, 분할 순차 스캔 방식에서는 제한된 IP 주소 범위를 여러 구역으로 나누고, 봇이 서로 다른 구역을 스캔하게 한다. 그래서 사용 가능 주소 범위를 나눠 만든 구역 수만큼 각 구역을 스캔할 봇이 필요하다. 분할 순차 스캔 방식은 제한된 IP 주소 범위의 구역을 나누기 위해 모듈러 산술을 사용한다.

즉, 봇은 호스트 ID를 구역 수로 나눗셈한 나머지 값으로 본인이 스캔할 IP 주소를 판별한다. 예를 들어 그림 7과 같이 사용 가능 주소 범위를 2구역으로 나눈다면, 한 봇은 호스트 ID를 2로 나눗셈했을 때 나머지 값이 1인 IP 주소만을 스캔하고 다른 봇은 나머지 값이 0인 IP 주소만을 스캔한다.

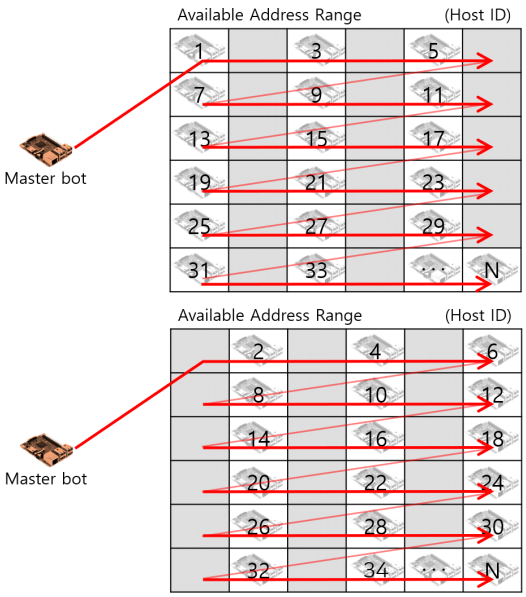

Fig. 7. Divided and sequential scanning method

4.2.1.3 제한 랜덤 스캔 방식

제한 랜덤 스캔 방식은 그림 8과 같이 제한된 IP 주소 범위에서 랜덤으로 IP 주소를 선택한다. 다른 스캔 방식처럼 봇의 로컬 네트워크 주소로 사용 가능 주소 범위를 얻는다.

미라이 악성코드 또한 랜덤 스캔 기법을 사용하지만, 스캔 범위에 제한이 없어 전 세계에 있는 취약한 IoT 기기에 접근할 수 있다. 그러나 제한 랜덤 스캔 방식은 봇이 접근할 수 있는 주어진 IP 주소 범위에서만 스캔할 수 있다. 제한 랜덤 스캔 방식은 접근할 수 있는 장치 수는 제한되지만 같은 네트워크에 존재하는 장치에 빠르게 접근할 수 있다는 장점이 있다.

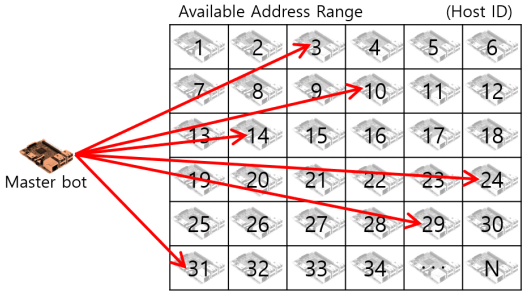

Fig. 8. Restricted random scanning method

4.2.1.4 분할 랜덤 스캔 방식

분할 랜덤 스캔 방식은 제한 랜덤 스캔 방식과 동일하게 제한된 IP 주소 범위에서 랜덤으로 스캔할 IP 주소를 선택한다. 그러나 분할 랜덤 스캔 방식은 제한된 IP 주소 범위를 여러 영역으로 나누고 봇이 서로 다른 영역을 스캔하도록 한다.

제한된 IP 주소 범위를 두 구역으로 나누면, 두 개의 봇은 그림 9과 같이 랜덤으로 서로 다른 구역을 스캔한다. 분할 순차 스캔 방식과 마찬가지로 봇은 모듈러 연산을 사용하여 구역을 구분한다.

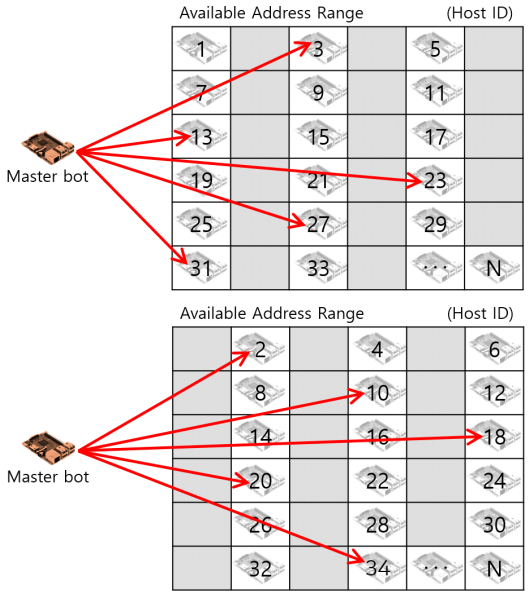

Fig. 9. Divided and random scanning method

4.2.2 포트 스캔 탐지 및 회피 기법

네트워크 스캔 기법으로는 호스트 스캔, 포트 스캔, 취약점 스캔이 있다. 미라이 악성코드는 취약한 기기를 찾기 위해 포트 스캔하였으며, 포트 스캔 중에서도 TCP SYN 스캔 기법을 사용하였다.

4.2.2.1 포트 스캔 탐지 기법

포트 스캔을 탐지하는 방법으로는 트래픽 속도 평가 또는 트래픽 모니터링을 통한 시그니처 및 룰 분석 기법, 이상 행위 분석 기법, 상태정보를 저장하는 스테이트풀(stateful) 프로토콜 분석 기법이 있다. 시

그니처 및 룰 분석 기법은 알려진 침입 방법과 비교하여 비정상적인 행위 형태를 보인 패킷을 탐지한다. 이상 행위 분석 기법은 일반적인 행위 패턴을 프로파일로 생성하여 프로파일 범위를 벗어난 패킷을 탐지한다. 앞서 설명한 두 기법 모두 수신한 패킷에 존재하는 정보를 사용하여 일반적인 패킷인지 스캔에 사용된 패킷인지 식별할 수 있다. 만약 스캔에 사용된 패킷이 미리 지정된 임계값만큼 식별되는 경우 스캔이 수행되고 있음을 탐지한다. 스테이트풀 프로토콜 분석 기법은 TCP NULL 스캔, TCP SYN/FIN 스캔, TCP FIN 스캔, TCP Xmas 스캔과 같이 TCP 연결을 맺는 데 일반적이지 않은 비정상 패킷을 탐지한다. TCP의 상태 변화로는 SYN을 전송하지 않은 상태에서는 SYN/ACK 패킷을 수신할 수 없다. TCP 연결을 위한 상태 정보 흐름을 분석하여 흐름이 이상한 패킷을 수신하면 스캔이 수행되고 있음을 탐지한다. 포트 스캔을 탐지하는 대표적인 프로그램으로 RTSD(Real Time Scan Detector), Snort, PortSentry, Scanlogd가 있다.

4.2.2.2 포트 스캔 탐지 기법 회피 방법

트래픽 속도로 스캔을 탐지하는 경우 느린 스캔을 수행하여 스캔 탐지를 회피한다. 또한, 봇넷에서 수행하는 스캔은 여러 개의 봇이 여러 개의 취약한 장치를 스캔하는 병렬화된 스캔이므로 스캔을 탐지하기 어렵다. 따라서 시그니처 및 룰 분석 기법으로 탐지하는 경우 임계값을 넘지 않을 정도로 스캔하면 스캔 탐지를 회피할 수 있다고 판단된다. 이상 행위 분석 기법으로 스캔을 탐지하는 경우 일반적인 호스트에 대한 지식을 수집하여 비적대적 스캔을 수행하면 스캔 탐지를 회피할 수 있다고 판단되며, 스테이트풀 프로토콜 분석 기법으로 탐지하는 경우 무상태 프로토콜을 사용하여 스캔하면 스캔 탐지를 회피할 수 있다고 판단된다. 또한, 포트 스캔을 탐지하는 탐지 시스템을 무력화하기 위해 탐지 시스템이 탐지할 수 있는 비정상 패킷을 의도적으로 서비스 거부 형태로 보내 탐지 시스템을 중단시키면 스캔 탐지를 회피할 수 있다고 판단된다.

4.2.3 D2D 커맨드 인젝션 확산 기법

악성코드 고속 확산 도구에서는 D2D(Device to Device) 커맨드 인젝션(Command Injection, CI) 확산 기법을 적용한다. D2D 커맨드 인젝션 확산 기법은 봇이 취약한 IoT 기기에 파일 전송 명령어를 주입하여 취약한 IoT 기기가 주입된 명령어를 실행해 서버로부터 악성코드 바이너리를 내려받는 방식이다.

미라이 악성코드의 로더 배포 악성코드 확산 기법을 사용하면 사설 IP 주소를 할당받은 내부 네트워크 내 IoT 기기에 내부 네트워크 밖에 있는 멀웨어 로더가 접근할 수 없어 멀웨어 로더가 IoT 기기에 악성코드를 배포할 수 없다. 멀웨어 로더가 내부 네트워크에 기기에 접속하려면 내부 네트워크 기기마다 포트 포워딩을 설정하거나, 멀웨어 로더가 공격 대상 기기와 같은 네트워크에 있어야 한다. 그렇지 않고서는 멀웨어 로더는 사설 IP 주소를 할당받은 내부 네트워크 기기에 접근할 수 없다. 미라이 악성코드가 내부 네트워크를 스캔 대상에서 제외한 것도 이와 같은 이유이다.

미라이 악성코드의 확산 기법이 가진 문제점을 개선하고자 악성코드 고속 확산 도구는 특정 IoT 인프라 내에서도 악성코드를 확산하기 위해 D2D 커맨드 인젝션 기법을 적용하였다. D2D 커맨드 인젝션 확산 방식에서는 미라이 악성코드에서 사용되었던 리포트 서버는 사용되지 않는다. 또한, 멀웨어 로더가 하던 작업을 봇이 대신한다. 즉, 악성코드 고속 확산 도구가 적용한 D2D 커맨드 인젝션 확산 방식에서는 봇이 취약한 기기에 파일 다운로드 명령을 주입하여 취약한 기기가 HTTP 서버에게 악성코드를 요청하도록 한다. 미라이 악성코드와 달리 악성코드 고속 확산 도구는 내부 네트워크 기기가 상위 네트워크에 있는 HTTP 서버에 접근하기 때문에 내부 네트워크에서도 악성코드 확산이 가능하다.

4.2.3.1 D2D 커맨드 인젝션 확산 기법 절차

악성코드 고속 확산 도구는 그림 10과 같이 총 6단계의 확산 과정을 거쳐 취약한 IoT 기기를 감염한다.

1단계. 텔넷 포트 스캔. 봇은 텔넷 서비스를 사용하는 기기를 스캔한다. 목적지 포트 번호를 23번 또는 2323번으로 설정해 패킷을 전송하고, 응답의 여부로 텔넷 서비스가 활성화되어 있는지 확인한다. 텔넷 포트 스캔 방식은 미라이 악성코드와 동일하다.

2단계. 텔넷 로그인을 위한 자격증명 무차별 대

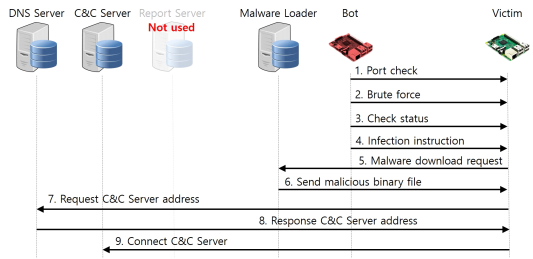

Fig. 10. The stages of the high-speed malware propagation attack tool

입 공격. 봇은 텔넷 서비스가 활성화된 기기를 대상으로 접속을 시도한다. 62개의 자격증명으로 최대 30초 동안 무차별 대입 공격을 한다.

3단계. 취약한 장치에 대한 정보 수집. 텔넷 로그인에 성공하면 취약한 기기에 악성코드를 주입하기 위한 사전 조사가 이뤄진다. 봇은 취약한 기기에 대한 CPU 아키텍처, 셸(shell) 종류 등 악성코드를 주입하는 데 필요한 정보를 수집한다.

4단계. 악성코드를 내려받는 명령어 주입. 봇은 취약한 기기가 악성코드를 내려받도록 명령어를 주입한다. 봇은 수집한 정보를 바탕으로 취약한 기기와 같은 아키텍처 기반의 악성코드 바이너리를 결정한다.

5단계. 악성코드 다운로드 요청. 취약한 기기는 봇이 주입한 명령어를 실행한다. 취약한 기기는 Wget 명령을 사용해 악성코드가 저장된 HTTP 서버이자 멀웨어 로더에 파일 다운로드를 요청한다.

6단계. 악성코드 주입. 멀웨어 로더는 취약한 기기의 요청에 응답해 악성코드를 주입한다.

4.3 SSS 성능 검증을 위한 문제점 해결방안

앞서 IoT 인프라에 확산하는 위협을 탐지하고 차단하는 SSS 보안 솔루션의 기능을 검증하기 위해 제한 범위 스캔 기법과 D2D 커맨드 인젝션 확산 기법을 제안하고 이 방식을 적용한 악성코드 고속 확산 도구를 구현하였다. SSS 보안 솔루션의 기능 검증에 이어서 SSS 보안 솔루션의 성능을 검증하기 위해 악성코드 고속 확산 도구는 악성코드를 IoT 인프라 내 빠르게 확산하고자 한다.

IoT 인프라 내 악성코드를 빠르게 확산하기 위해서는 감염되지 않은 취약한 기기를 빠르게 스캔하고 중복 감염을 최소화해야 한다. 감염되지 않은 취약한 기기를 스캔하고자 스캔 패킷 개수[1]와 마스터 봇[2] 개수를 임계치보다 높게 설정하면 중복 감염이 발생하여 악성코드 확산하는 양상을 보일 수 없다. 반대로 중복 감염을 줄이고자 스캔 패킷 개수와 마스터 봇 개수를 임계치보다 낮게 설정하면 IoT 인프라 내 악성코드가 빠르게 확산하는 양상을 보일 수 없다. 즉, IoT 인프라 내 악성코드 고속 확산을 위해서는 인프라 크기와 구성 기기 수를 고려하여 중복 감염이 발생하지 않을 정도의 마스터 봇의 개수와 스캔 패킷 개수로 설정해야 한다.

V. 악성코드 고속 확산을 위한 마스터 봇 및 스캔 패킷 변화 실험 및 평가

악성코드 고속 확산 도구의 확산 속도, 효율성 및 정확도를 높이기 위한 최적의 조건을 찾기 위해 스캔 방식, 마스터 봇 개수, 스캔 패킷 개수를 변경하여 내부 네트워크에서 악성코드 확산 실험을 수행하였다.

5.1 실험 환경

그림 11, 12와 같이 악성코드 확산 실험을 위해 실제 환경과 가상 환경을 구축하였다.

5.1.1 실제 환경

그림 11과 같이 실제 환경에서는 내부 네트워크 내 악성코드 확산 실험을 위한 C&C 서버, DNS

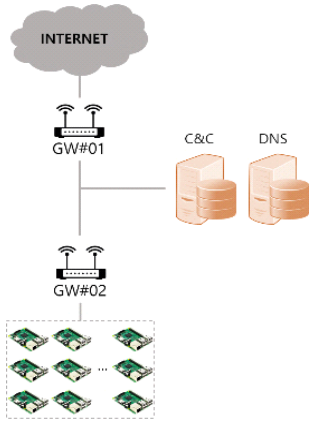

Fig. 11. The real world experiment environment

1) 마스터 봇이 포트 스캔 1회당 내보내는 전체 패킷
2) 스캔 동작을 수행하는 봇

Table 2. Specification of devices used in the real world environment

Server	CPU	Intel(R) Core(TM) i7 CPU 960 @ 3.20GHz
	RAM	12G
	OS	Ubuntu bionic Ubuntu 18.04.5 LTS)
Gateway	CPU	ARMv7 Processor 5(v7l)
	RAM	497MB
	OS	OpenWrt SNAPSHOT r13968-c5360894dc
IoT device	CPU	ARMv7 Processor 4 (v7l)
	RAM	1GB
	OS	Raspbian GNU/Linux 9 stretch

서버와 같은 봇넷 인프라가 1번 게이트웨이에 연결되어 있으며, 2번 게이트웨이에는 45대의 라즈베리 파이가 연결되어 있다.

실제 환경에서 사용한 서버, 게이트웨이, IoT 기기의 규격은 표 2와 같고, 그중 게이트웨이는 본 연구실에서 개발한 위협 확산 방지 기술이 탑재된 네트워크 장비이다.

5.1.2 가상 환경

그림 12와 같이 가상 환경은 QEMU를 사용해 단일 호스트 PC 내에서 C&C 서버, DNS 서버,

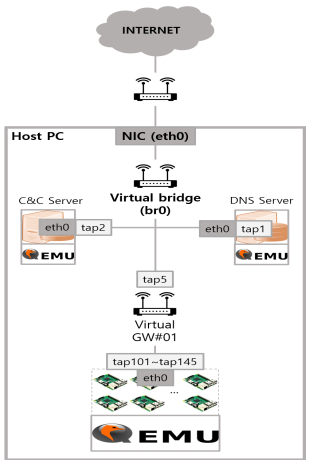

Fig. 12. The virtual world experiment environment

Table 3. Specification of devices used in the virtual world environment

Host PC	CPU	Intel(R) Core(TM) i9-10900 CPU @ 2.80GHz, 20 Core
	RAM	128GB
	OS	Ubuntu bionic Ubuntu 18.04.5 LTS)
Emulated Server	CPU	Intel(R) Core(TM) i9-10900 CPU @ 2.80GHz, 4 Core
	RAM	8GB
	OS	Ubuntu bionic Ubuntu 18.04.5 LTS)
Emulated IoT device	CPU	ARM1176(ARMv6-compatible processor rev 7 (v6l)
	RAM	253MB
	OS	Raspbian GNU/Linux 9 stretch

그리고 45대의 라즈베리 파이를 에뮬레이터 하였다. 가상 네트워크 환경을 구축해 그림 11의 실제 환경과 동일하게 구현하였다. 또한, 에뮬레이션 된 가상 기기끼리 서로 통신이 가능하도록 사설 IP 주소가 할당되어 있다.

가상 환경에서 사용한 호스트 PC, 에뮬레이션 된 서버와 IoT 기기의 규격은 표 3과 같다. QEMU 에뮬레이터 버전 2.11.1을 사용해 C&C 서버, DNS 서버, 라즈베리 파이를 에뮬레이션하였으며, IoT 기기의 경우 에뮬레이션을 위해 2018-06-27 Raspbian stretch lite OS 이미지 파일과 4.14.79-stretch QEMU 커널을 사용하였다.

5.2 성능 분석

IoT 인프라에 빠르게 퍼지는 위협을 탐지하고 차단하는 SSS 보안 솔루션의 기능과 성능을 검증하기 위하여 IoT 인프라 내 악성코드를 빠르게 확산하여야 한다. 네트워크 장비에 과부하를 줄이고 중복 감염이 발생하지 않으면 IoT 인프라 내 악성코드를 빠르게 확산할 수 있다고 판단하였으며, 과부하와 중복 감염에 영향을 미치는 요인을 스캔 패킷 개수와 마스터 봇 개수라고 판단하였다. 이에 따라 IoT 인프라 크기를 고려한 스캔 패킷 개수, 마스터 봇 개수, 그리고 스캔 방식을 찾기 위한 확산 실험을 수행하였다.

5.2.1 스캔 패킷 개수에 따른 악성코드 확산 성능

악성코드 확산의 속도와 안정성이 스캔 패킷 개수에 영향을 받는다고 판단하여 스캔 패킷 수에 따른 악성코드 확산 실험을 수행하였고, 제안된 스캔 방식별로 총 확산 시간과 중복 감염 수를 측정하였다. IoT 기기가 C&C 서버에 연결되면 악성코드에 감염된 것으로 판단하여 실험에 사용된 45대의 모든 기기가 C&C 서버에 연결되었을 때, 첫 번째 연결 시간과 마지막 연결 시간의 차이를 계산하여 총 확산 시간을 측정하였다. 중복 감염 수의 경우 HTTP 서버이자 멀웨어 로더가 IoT 기기로부터 요청받은 전체 악성코드 다운로드 요청 개수에서 모든 IoT 기기를 감염시키기 위해 최소한으로 필요한 45개의 다운로드 요청 수를 제외하여 측정하였다. 또한, 스캔 패킷 개수가 총 확산 시간과 중복 감염에 미치는 영향을 확인하기 위해 하나의 마스터 봇만 사용하여 악성코드 확산 실험을 수행하였다.

그림 13과 그림 15는 실제 환경에서 스캔 패킷 개수에 따른 스캔 방식별 악성코드 확산 실험 결과를 나타낸 그래프이고, 그림 14와 그림 16은 가상 환경에서 스캔 패킷 개수에 따른 스캔 방식별 악성코드 확산 실험 결과를 나타낸 그래프이다. 그림 13부터 그림 16까지는 악성코드 확산 실험 시 스캔 패킷의 수를 최소 1개부터 5의 배수로 증가하여 최대 45개까지 설정하여 악성코드 확산 실험을 수행한 결과이다. 스캔 패킷 개수에 따른 악성코드 확산 실험 수행 시 스캔 패킷 개수를 1부터 45개까지 설정한 이유는 스캔 패킷 개수가 45개 이상으로 설정되면 총 확산 시간이 눈에 띄게 줄거나 증가하지 않았다. 또한, 6

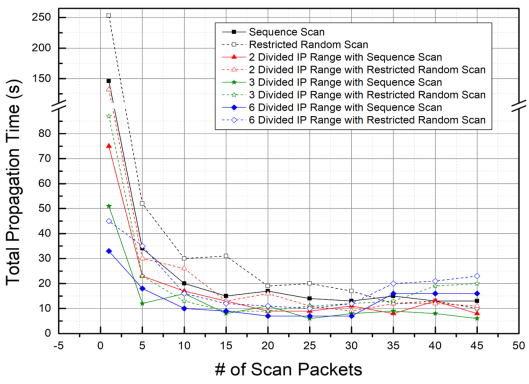

Fig. 13. Total propagation time according to the number of scan packets in the real world

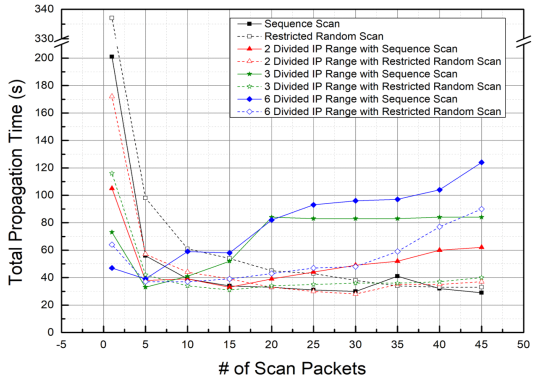

Fig. 14. Total propagation time according to the number of scan packets in the virtual world

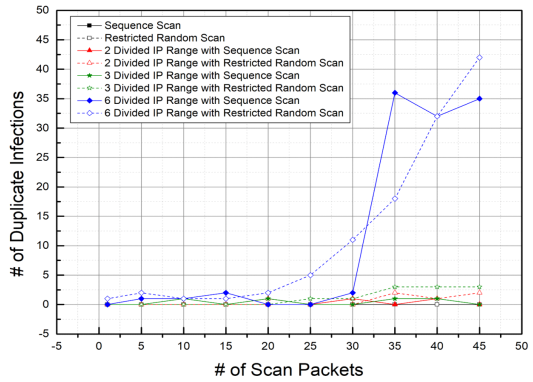

Fig. 15. The number of duplicate infections according to the number of scan packets in the real world

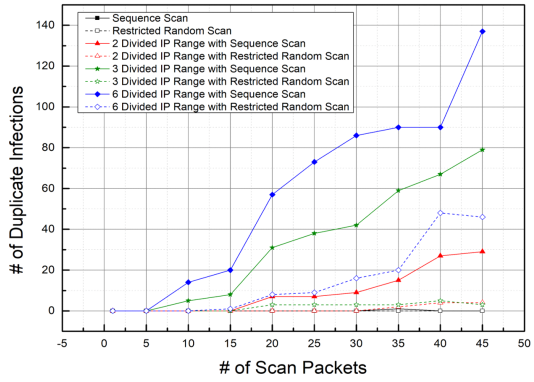

Fig. 16. The number of duplicate infections according to the number of scan packets in the virtual world

분할 스캔 방식에서 스캔 패킷 수가 45개를 넘어서면 중복 감염이 급증하여 실험에 사용된 기기 45대

를 전체적으로 감염시키지 않는 문제가 발생하였다.

그림 13과 그림 14는 스캔 패킷 개수에 따른 스캔 방식별 총 확산 시간을 보여주고 있다. 분할된 단계로 그룹 지어 순차 스캔 방식과 랜덤 스캔 방식의 총 확산 시간을 비교한 결과 순차 스캔 방식을 사용하는 것이 랜덤 스캔 방식을 사용하는 것에 비해 총 확산 시간을 비교적 단축할 수 있었다. 즉, 순차 스캔 방식과 제한 랜덤 스캔 방식을 비교했을 때, 순차 스캔 방식을 악성코드 확산 실험에 적용하면 총 확산 시간이 더 짧았다. 이러한 결과는 2분할 스캔 방식(2분할 순차 스캔 방식과 2분할 랜덤 스캔 방식), 3분할 스캔 방식, 그리고 6분할 스캔 방식에서도 동일하게 확인되었다. 다만 순차 스캔 방식과 제한 랜덤 스캔 방식에서 스캔 패킷 개수가 35개 이상일 때, 결과가 뒤집혔지만 두 스캔 방식에 대한 총 확산 시간의 격차가 커지지 않았고, 거의 비슷한 시간을 유지하였다. 따라서 IP 주소 범위가 제한된 내부 네트워크와 같은 환경에서는 랜덤 스캔 방식보다 순차 스캔 방식을 사용하는 것이 총 확산 시간을 줄이는 데 효과적이다.

스캔 패킷 개수가 증가함에 따라 총 확산 시간이 줄어들었으나, 스캔 패킷 개수가 일정 개수를 초과하면 총 확산 시간이 줄어들지 않고 일정한 시간을 유지하였다. 특히, 분할 스캔 방식과 같이 스캔할 IP 주소 범위가 좁은 환경에서는 오히려 스캔 패킷 수가 증가함에 따라 총 확산 시간이 증가하였다. 이러한 결과가 나타나는 이유는 스캔할 IP 주소 범위가 좁은 환경에서 스캔 패킷 개수가 적절한 수보다 높은 경우 여러 마스터 봇이 같은 취약한 기기를 스캔할 수 있고, 이로 인해 중복 감염이 발생하기 때문이다. 실제로 스캔 패킷 개수를 똑같이 하여 악성코드 확산 실험을 수행해도 2분할, 3분할, 6분할 스캔 방식을 비교해보면 6분할 스캔 방식의 총 확산 시간이 가장 오래 걸렸다. 따라서 스캔할 IP 주소 범위가 좁은 내부 네트워크 환경에서는 10개에서 20개 사이의 적절한 개수의 스캔 패킷을 사용하면 총 확산 시간을 줄일 수 있다. 즉, 최소한의 스캔 패킷으로 짧은 시간에 내부 네트워크의 전체 기기를 감염시킬 수 있다.

결과적으로 미라이 악성코드가 사용하는 랜덤 스캔 방법은 전 세계의 취약한 장치를 검색하고 감염시키는 데 적합하지만, 순차 스캔 방법은 IP 주소 범위가 제한된 내부 네트워크 환경에서 적합하다. 특히 실험 환경과 같은 환경에서는 2분할 스캔 방법과 15개 스캔 패킷을 사용하면 내부 네트워크에 악성코드를 빠르게 전파할 수 있다. 즉, 확산 실험 대상 환경에 따라 적절한 스캔 방식과 스캔 패킷 개수를 선택하는 것이 총 확산 시간을 줄이는 데 효과적이다.

그림 15와 그림 16은 실제 환경과 가상 환경에서 스캔 패킷 개수에 따른 스캔 방식별 중복 감염 개수를 보여주고 있다. 두 그래프 모두 스캔 패킷 개수가 일정 개수를 넘어서면 중복 감염 횟수가 증가하였다. 순차 스캔 방식과 제한 랜덤 스캔 방식에서는 스캔 패킷 개수가 증가해도 중복 감염 수가 큰 폭으로 증가하지 않았다. 하지만 분할 스캔 방식에서는 스캔 패킷 수가 특정수를 초과하면 중복 감염 수가 증가하였다. 특히 6분할 순차 스캔 방식과 6분할 랜덤 스캔 방식에서 스캔 패킷 개수가 늘면서 중복 감염 수가 급증하였다. 이는 제안된 스캔 방법 중에 가장 작은 IP 주소 범위를 스캔하기 때문이다. 스캔 가능한 IP 주소 범위가 좁은 데 비해 스캔 패킷 수가 크면 하나의 마스터 봇에서 내보낸 여러 스캔 패킷이 동시에 같은 IP 주소를 스캔할 가능성이 크다. 취약한 장치가 이미 감염됐더라도 C&C 서버에 연결될 때까지 계속 감염될 수 있으므로, 완전히 감염되기 전에 취약한 기기가 여러 번 스캔 되면 중복 감염이 발생한다. 그림 16의 그래프는 이러한 결과를 잘 보여준다. 스캔 패킷 개수를 동일하게 설정해도 중복 감염 수는 6분할 스캔 방식이 가장 높았다.

그림 15와 그림 16을 통해 실제 환경과 가상 환경 모두 스캔 패킷 개수가 일정 수를 넘어서면 중복 감염이 많이 발생하는 것을 확인하였다. 그러나 중복 감염에 따른 실제 환경과 가상 환경 결과를 비교하면 가상 환경일 때 중복 감염 수가 더 많았다. 이는 가상 환경에서는 가상 머신을 구동하는 호스트 PC의 낮은 하드웨어 성능으로 인한 것으로 보인다. 하드웨어 성능이 실제 네트워크 장비만큼 지원되지 않아 감염 속도가 스캔 속도보다 느려 중복 감염이 발생한 것으로 보인다. 스캔 속도는 마스터 봇이 취약한 기기를 찾는 속도를 의미하며, 감염 속도는 취약한 기기가 악성코드를 서버로부터 내려받아 악성코드를 실행하고 C&C 서버에 접속하는 속도를 의미한다. 가상 환경에서는 호스트 PC의 제한적인 하드웨어 성능으로 인해 취약한 기기가 악성코드를 서버로부터 내려받는데 실제 환경보다 더 오랜 시간이 필요한 것으로 보인다. 따라서 가상 환경에서는 스캔 속도보다 감염 속도가 느리므로 스캔 패킷 개수가 적더라도 많은 중복 감염이 발생한 것으로 보인다.

실제 환경에 대한 그림 13의 그래프를 보며 2분

할 순차 스캔 방식을 사용하는 것이 총 확산 시간을 줄이는 데 효과적이라는 것을 확인하였다. 그러나 가상 환경에 대한 그림 14의 그래프에서는 분할 순차 스캔 방식이 분할 랜덤 스캔 방식보다 총 확산 시간이 더 오래 걸렸다. 이러한 이유는 그림 16의 그래프와 같이 악성코드 확산 실험에서 분할 순차 스캔 방식을 사용했을 때 가상 환경에서 중복 감염의 수가 많기 때문이다. 순차 스캔 방식과 랜덤 스캔 방식을 비교했을 때 순차 스캔 방식에서 중복 감염이 많이 발생한 이유는 스캔 속도의 차이 때문이다. 순차 스캔 방식의 경우 스캔 가능한 IP 주소 범위의 처음부터 끝까지 호스트 ID만 1씩 증가하면 되기 때문에 IP 주소를 빠르게 선택할 수 있다. 반면에 랜덤 스캔 방식은 스캔 가능한 IP 주소 범위에서 무작위로 하나를 선택하지만 주소 범위 이내의 모든 IP 주소가 한 번씩 스캔에 사용되기 전에는 이전에 사용된 IP 주소를 다시 또 사용할 수 없다. 그래서 랜덤 스캔 방식에서는 선택된 IP 주소가 이전에 사용되었는지 확인하는 절차를 거치게 되고, 이 때문에 스캔 속도에 약간의 지연이 발생한다. 즉, 순차 스캔 방식은 IP 주소를 빠르게 선택할 수 있어 랜덤 스캔 방식보다는 상대적으로 스캔 속도가 빠르다. 특히 6분할 스캔 방식처럼 스캔할 IP 주소 범위가 좁은 경우 전체 IP 주소 범위를 처음부터 끝까지 스캔하는 시간 간격이 매우 좁다. 그래서 순차 스캔 방식에서는 랜덤 스캔 방식보다 스캔 속도가 더 빨라 기기가 완전히 감염되기 전에 다시 스캔 되어 중복 감염이 많이 발생하였다.

그림 13부터 그림 16을 통해 실제 환경과 가상 환경 모두 중복 감염에 따라 총 확산 시간이 영향을 받는 것을 확인하였다. 두 환경 모두 동일한 경향을 나타내므로 그림 17부터 그림 20까지는 오직 실제 환경에서의 스캔 패킷 개수에 따른 총 확산 시간과 중복 감염 횟수를 비교하였다.

그림 17은 스캔 패킷 개수에 따른 순차 스캔 방식과 제한 랜덤 스캔 방식에 대한 총 확산 시간과 중복 감염에 대한 그래프이다. 두 스캔 방식 모두 내부 네트워크의 모든 IP 주소를 스캔하므로 스캔 가능한 IP 주소 범위가 넓다. 그래서 스캔 패킷 개수를 늘리면 총 확산 시간이 단축되고 중복 감염이 발생하지 않는다. 스캔할 IP 주소 범위가 넓어 중복 감염을 발생시키지 않았기 때문에, 그림 17의 그래프에서 볼 수 있듯이, 총 확산 시간은 스캔 패킷 수가 증가함에 따라 감소했고 특정 스캔 패킷 수를 초과하면서

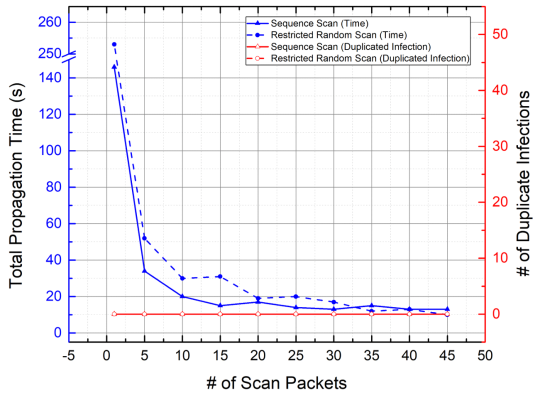

Fig. 17. Total propagation time and the number of duplicate infections according to the number of scan packets in the sequential scanning method and the restricted random scanning method

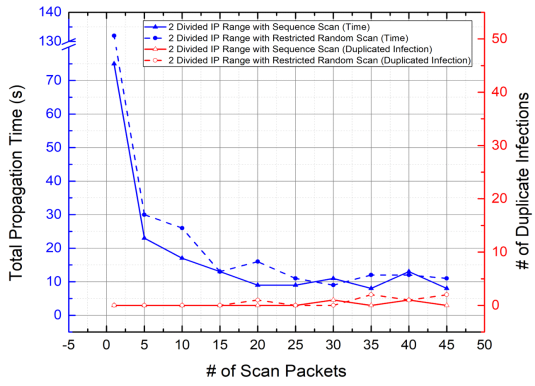

Fig. 18. Total propagation time and the number of duplicate infections according to the number of scan packets in the 2-divided and sequential scanning method and the 2-divided and random scanning method

일정하게 유지되었다. 그림 18은 스캔 패킷 개수에 따른 2분할 순차 스캔 방식과 2분할 랜덤 스캔 방식에 대한 총 확산 시간과 중복 감염에 대한 그래프이다. 2분할 스캔 방식은 내부 네트워크의 전체 IP 주소 범위를 두 영역으로 나눈다, 그래서 두 개의 마스터 봇은 자신이 속한 영역의 IP 주소만 스캔한다. 스캔 가능 IP 주소 범위는 순차 검색 방법과 제한 랜덤 스캔 방법에 비해 좁아졌지만, 다른 분할 스캔 방식에 비하면 상대적으로 넓다. 그래서 2분할 스캔 방식에서는 스캔 패킷 개수가 25개 이상일 때 중복 감염되는 모습을 보인다. 2분할 랜덤 스캔 방식의 경우 중복 감염이 발생한 스캔 패킷 20, 35, 45개

에서 총 확산 시간이 증가하였고, 2분할 순차 스캔 방식의 경우 중복 감염이 발생한 스캔 패킷 30, 40개에서 총 확산 시간이 증가하였다. 두 그래프를 볼 때, 중복 감염은 총 확산 시간에 영향을 미치며, 최소한의 스캔 패킷으로 총 확산 시간을 단축하기 위해선 2분할 순차 스캔 방식을 사용하는 것이 효율적이다. 그림 19는 스캔 패킷 개수에 따른 3분할 순차 스캔 방식과 3분할 랜덤 스캔 방식에 대한 총 확산 시간과 중복 감염 수를 보여준다. 3분할 스캔 방식은 내부 네트워크의 사용 가능한 IP 주소 범위를 3구역으로 나눈다. 그래서 마스터 봇이 스캔할 수 있는 IP 주소 범위가 2분할 스캔 방식보다 좁다. 그 결과, 그림 19와 같이 적은 스캔 패킷 개수에서도 중복 감염이 발생하였다. 또한 중복 감염이 발생할 때 총 확산 시간이 증가했으며 이는 2분할 스캔 방식과 같았다. 3분할 랜덤 스캔 방법의 경우 중복 감염이 있는 20개 이상의 스캔 패킷에서 총 확산 시간이 증가했으며, 3분할 순차 스캔 방식의 경우, 총 확산 시간은 중복 감염이 있는 10, 20, 35, 40개의 스캔 패킷에서 증가하였다. 그림 20은 스캔 패킷 개수에 따른 6분할 순차 스캔 방식과 6분할 랜덤 스캔 방식에 대한 총 확산 시간과 중복 감염에 대한 그래프이다. 마스터 봇이 스캔할 IP 주소 범위가 제안한 스캔 방식 중 가장 좁다. 그래서 스캔 패킷 개수가 증가할수록 중복 감염 횟수가 급증하였다. 또한, 이전 스캔 방식과 동일하게 중복 감염이 발생한 부분에서 총 확산 시간이 증가하였다. 특히 스캔 패킷 30개 이후로 중복 감염 발생 횟수가 급증했고 총 확산 시간도 많이 증가하였다.

그림 13과 그림 14를 통해 순차 스캔 방식을 사용하면 랜덤 스캔 방식보다 더 적은 수의 스캔 패킷을 사용해도 총 확산 시간을 줄일 수 있다는 것을 확인하였다. 그러나 스캔 패킷 개수가 증가해 일정 수를 넘어서면 총 확산 시간이 일정하게 유지되거나 증가하였다. 또한, 중복 감염은 총 확산 시간에 영향을 미쳐, 중복 감염이 발생하면 총 확산 시간이 더 오래 걸렸다. 그림 15와 그림 16은 스캔 패킷 개수가 특정수보다 많은 경우 중복 감염이 발생했음을 보여주며, 이러한 경향은 분할 스캔 방식처럼 스캔할 IP 주소의 범위가 좁은 환경에서 더욱 두드러진다. 또한, 그림 15와 그림 16에서는 실제 환경보다 가상 환경에서 중복 감염이 더 많이 발생하였는데, 이는 가상 머신을 구동하는 호스트 PC의 낮은 하드웨어 성능으로 인해 스캔 속도보다 감염 속도가 느려 발생

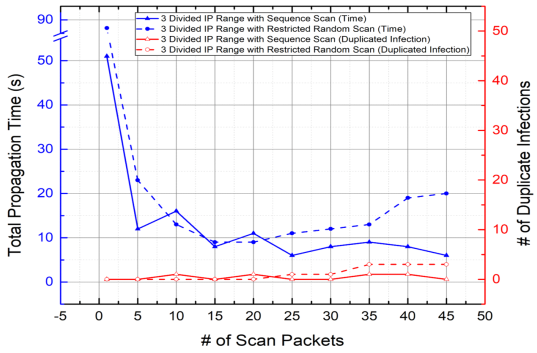

Fig. 19. Total propagation time and the number of duplicate infections according to the number of scan packets in the 3-divided and sequential scanning method and the 3-divided and random scanning method

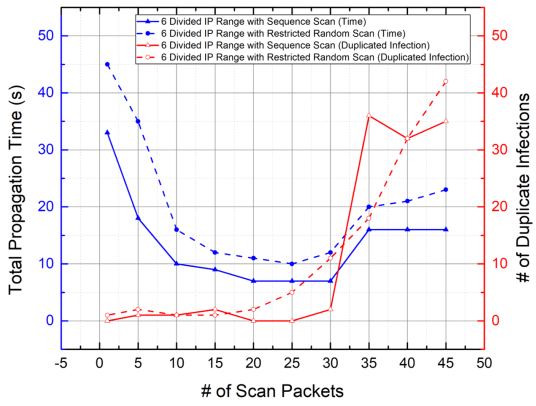

Fig. 20. Total propagation time and the number of duplicate infections according to the number of scan packets in the 6-divided and sequential scanning method and the 6-divided and random scanning method

한 것으로 보인다. 게다가, 가상 환경에서 순차 스캔 방식은 랜덤 스캔 방식보다 더 많은 중복 감염이 발생하였는데, 이는 두 스캔 방식의 스캔 속도 차이 때문으로 보인다. 이러한 중복 감염의 위험은 스캔 가능한 IP 주소 범위가 여러 영역으로 분할될 때 증가하였다. 다음으로, 그림 17에서 그림 20 그래프를 통해 중복 감염이 총 확산 시간에 영향을 미친다는 것을 확인하였다. 따라서 빠르고 정확한 악성코드 확산을 수행하여야 하는 경우 순차 스캔 방법 중 중복 감염 횟수가 가장 적은 스캔 패킷 수와 총 확산 시간이 가장 짧은 스캔 방식을 선택하는 것이 가장 효율적이다.

VI. 결 론

IoT 환경에서는 수많은 보안 위협을 방지하기 위한 다양한 보안 솔루션이 존재한다. 공격 대상과 방법이 다양해 보안 솔루션 하나로 막을 수 없기 때문이다. 수많은 보안 솔루션 중에 실제 환경에서 필요한 보안 솔루션을 적절하게 선택하기 위해서는 보안 솔루션의 기능 및 성능에 대한 검증이 필요하다. 보안 솔루션의 작동 방식과 보호 대상이 무엇인지 알아야 보안 솔루션에 대한 정확한 기능과 성능 시험이 가능하다. 여러 보안 솔루션 중 위협 확산 방지 기술의 스마트 세그멘테이션 솔루션(SSS)은 IoT 인프라 내 위협을 빠르게 탐지하고 차단하는 보안 솔루션이다. SSS 보안 솔루션은 네트워크 패킷을 분석하여 위협을 탐지하고 위험도에 따라 관리 대상의 네트워크 연결을 제어한다. SSS 보안 솔루션이 실제 공격에서도 위협을 탐지하고 차단하는지 확인하기 위해선 IoT 인프라에 위협을 확산하기 위해 IoT 인프라 내 악성코드를 확산하는 악성코드 고속 확산 도구를 구현하였다. 악성코드 고속 확산 도구는 인터넷에 공개된 미라이 악성코드를 기반으로 구현되었지만, 미라이 악성코드의 공격 대상과 공격 범위가 전 세계 IoT 기기를 대상으로 하여 특정 인프라 내 네트워크에 빠르게 확산하는 위협을 탐지하고 차단하는 SSS 보안 솔루션의 기능을 시험하는 데 적합하지 않았다. 따라서 SSS 보안 솔루션의 기능 시험을 위해 내부 네트워크 기기에 악성코드를 확산할 수 있는 제한 범위 스캔 기법과 D2D 커맨드 인젝션 확산 기법을 적용한 악성코드 고속 확산 도구를 구현하였다. 또한, SSS 보안 솔루션의 성능 시험을 위해 스캔할 IP 주소 범위가 제한된 환경에서 악성코드 확산 실험을 수행할 때 네트워크 장비에 부하를 주지 않으면서 빠르게 확산할 수 있는 조건을 찾고자 하였다. 이를 위해 악성코드 고속 확산 도구는 네 가지 스캔 방식을 구현해 스캔 방식에 따른 확산 속도를 비교했으며, 각 스캔 방식별로 마스터 봇의 개수와 스캔 패킷 개수를 변경하며 악성코드 확산 실험을 수행하였다.

실험 결과 총 확산 시간에 가장 큰 영향을 미친 것은 스캔 패킷의 개수였다. 스캔 패킷 수를 늘리면 총 확산 시간이 단축된 것을 확인하였다. 그러나 총 확산 시간을 줄이고자 스캔 패킷을 증가해도 일정 개수 이상부터는 총 확산 시간이 줄어들지 않았다. 이는 스캔 패킷의 개수가 증가하면 중복 감염 수가 증가하기 때문이다. 취약한 기기가 악성코드를 여러 번 내려받으면 킬러 프로세스에 의해 악성코드가 서로 동작하지 못한다. 그래서 중복 감염 횟수가 증가하면 감염 정확도가 줄어든다. 결과적으로 중복되는 패킷을 줄이기 위해 특정 스캔 패킷 수의 순차 스캔 방식을 사용하는 것이 가장 효율적이나, 이보다 좀 더 빠른 악성코드 확산을 원하는 경우 중복되더라도 분할 순차 스캔 방식을 사용하는 것이 가장 효율적임을 알 수 있다. 반대로 안정적이고 정확도 높은 악성코드 확산이 필요한 경우 속도는 조금 느리더라도 중복 감염 수가 적은 스캔 패킷 개수를 사용하는 것이 가장 효율적임을 알 수 있다. SSS 솔루션의 경우 IoT 인프라에 퍼지는 위협을 빠르게 탐지하고 차단하는 보안 솔루션임으로 SSS 솔루션의 기능을 시험하기 위해 빠른 악성코드 확산이 필요하였다. 그러므로 SSS 솔루션 기술을 설계하고 구현한 후에 분할 순차 스캔 방식으로 사용하여 시험함으로써 기능과 성능을 검증할 수 있었다.

본 논문에서 다루는 악성코드 고속 확산 도구는 SSS 보안 솔루션의 검증을 위해 구현되었지만, IoT 인프라에 네트워크 위협을 탐지 또는 방지하는 다른 보안 솔루션을 검증하는 목적으로도 사용할 수 있다. 또한, 현재까지는 악성코드 고속 확산 도구로 악성코드 확산만 가능하지만, 악성코드 고속 확산 도구는 미라이 악성코드를 기반으로 구현됐기 때문에 추후 연구 및 기능 구현을 통해 DDoS 공격 도구로도 사용할 수 있다. 이는 DPS 보안 솔루션을 포함하여 DDoS 공격을 탐지하거나 방지하는 네트워크 보안 솔루션을 검증하는 데 사용될 수 있다. 즉, 악성코드 고속 확산 도구에 봇넷을 활용한 공격 기능을 확장하면 여러 보안 솔루션을 실제 및 가상 시험망에서 검증할 수 있는 통합 검증 도구로 사용할 수 있다. 따라서 본 논문의 연구 결과를 바탕으로 추후 세 가지의 연구를 중점적으로 할 예정이다. 첫째는 본 논문에서 제안하는 악성코드 고속 확산 도구는 SSS 보안 솔루션에 최적화되어 있으나, 이후 DPS와 RAS 보안 솔루션에도 적용할 수 있는 스캔 및 공격 기법을 연구하여 통합 검증 도구를 구현하겠다. 둘째는 SSS 보안 솔루션을 위해 검증 도구를 구현하였으나, 아직은 자동화되어 있지 않아 스캔 패킷 개수와 마스터 봇 개수, 그리고 스캔 방식 설정 변경 시 매번 코드에서 직접 수정하여야 하는 불편함이 있다. 따라서, 스캔 방식과 스캔 패킷 수, 마스터 봇 개수 등을 GUI를 통해 설정할 수 있는 자동화된 검증 도구를 구현하겠다. 셋째로 본 논문에서 제안하고 있는

네트워크 위협 방지 솔루션 검증 도구에 IP 스캔 탐지를 회피하는 스캔 및 공격 기법을 추후 구현하여 IP 스캔 탐지 회피가 가능하면서 고속 확산도 가능한 악성코드 고속 확산 도구를 제작하고 이를 검증하겠다.

References

[1] M. Hatton and G. Chua, Service Provider Opportunities & Strategies in the Internet of Things, Machina Research, pp. 1-33, Dec. 2015.

[2] Y.B. Zikria, S.W. Kim, O. Hahm, MK. Afzal and MY. Aalsalem, "Internet of Things (IoT) Operating Systems Management: Opportunities, Challenges, and Solution," Sensors 2019, vol. 19, no. 8: 1793, pp. 1-10, Apr. 2019.

[3] M. Antonakakis, T. April, M. Bailey, M. Bernhard, E. Bursztein, J. Cochran, Z. Durumeric, J.A. Halderman, L. Invernizzi, M. Kallitsis, D. Kumar, C. Lever, Z. Ma, J. Mason, D. Menscher, C. Seaman, N. Sullivan, K. Thomas and Y. Zhou, "Understanding the Mirai Botnet," 26th USENIX Security Symposium (USENIX Security 17), pp. 1093-1110, Aug. 2017.

[4] GitHub, Inc., "mirai source code" https://github.com/jgamblin/Mirai-Source-Code, Sep. 28. 2019.

[5] SonicWall, "2020 SONICWALL CYBER THREAT REPORT: THREAT ACTORS PIVOT TOWARD MORE TARGETED ATTACKS, EVASIVE EXPLOITS" https://www.sonicwall.com/news/2020-sonicwall-cyber-threat-report/, Feb. 4. 2021.

[6] Palo Alto Networks, "2020 Unit 42 IoT Threat Report" https://unit42.paloaltonetworks.com/iot-threat-report-2020/, Feb. 10. 2021.

[7] IoT Analytics, "State of the IoT 2020: 12 billion IoT connections, surpassing non-IoT for the first time" https://iot-analytics.com/state-of-the-iot-2020-12-billion-iot-connections-surpassing-non-iot-for-the-first-time/, Feb. 19. 2021.

[8] Statista, "Internet of Things - active connections worldwide 2015-2025" https://www.statista.com/statistics/1101442/iot-number-of-connected-devices-worldwide/, Feb. 29. 2021.

[9] F. Hategekimana, T.J. Whitaker, M. J.H. Pantho and C. Bobda, "IoT Device security through dynamic hardware isolation with cloud-Based update," Journal of Systems Architecture, vol. 109, no. 1: 101827, pp. 1-11 Oct. 2020.

[10] B.B. Zarpelão, R.S. Miani, C.T. Kawakani and S.C. Alvarenga, "A survey of intrusion detection in Internet of Things," Journal of Network and Computer Applications, vol. 84, no. 1, pp.25-37, Apr. 2017.

[11] A. Saeed, A. Ahmadinia, A. Javed, H. Larijani, "IIntelligent Intrusion Detection in Low-Power IoTs," ACM Transactions on Internet Technology, vol. 16, no. 4: 27, pp. 1-25, Dec. 2016.

[12] Ministry of Science and ICT, "Wireless Communication Service Subscriber Statistics" https://www.msit.go.kr/SYNAP/skin/doc.html?fn=fdb04f93dec854f8856af3ef7c8b57fb&rs=/SYNAP/sn3hcv/result/, Feb. 26. 2021.

[13] 5G Americas, "the future of IoT" https://www.5gamericas.org/wp-content/uploads/2019/07/5G_Americas_White_Paper_on_5G_IOT_FINAL_7.16.pdf, Feb. 16. 2021.

[14] SonicWall, "2020 SONICWALL GLOBAL CYBER THREAT REPORT" https://w

ww.sonicwall.com/resources/2020-cyber-threat-report-mid-year-update-pdf/, Feb. 20. 2021.

[15] B. Vignau, R. Khoury and S. Hallé, "10 Years of IoT Malware: A Feature-Based Taxonomy," 2019 IEEE 19th International Conference on Software Quality, Reliability and Security Companion (QRS-C), pp. 458-465, Jul. 2019 .

[16] Korea Network Information Center, "Overall IPv4 address allocation status" https://xn—3e0bx5euxnjje69i70af08bea817g.xn--3e0b707e/jsp/statboard/IPAS/ovrse/total/currentV4Addr.jsp, Mar. 2. 2021.

〈저자소개〉

황 송 이 (Song-yi Hwang) 학생회원
2018년 8월: 백석대학교 정보보호학과 졸업
2019년 9월~현재: 과학기술연합대학원대학교 정보보호공학과 석사과정
〈관심분야〉 IoT 보안, 악성코드 분석, 네트워크 가상화, 암호화, 정보보호 등

김 정 녀 (Jeong-Nyeo Kim) 종신회원
1987년 2월: 전남대학교 전산통계학과 졸업
2000년 2월: 충남대학교 컴퓨터공학과 석사, 박사
1988년~현재: 한국전자통신연구원 정보보호연구본부 책임연구원
1996년: OSF/RI 공동연구 파견(미국)
2005년: Univ. of California, Irvine Post-Doc.
2015년~ 현재: 과학기술연합대학원대학교(UST) ICT(정보보호공학)과 교수
〈관심분야〉 IoT 보안, 모바일 보안, 시스템·네트워크 보안, 보안 OS 등